引用文献2

(19)日本国特許庁（ＪＰ）　(12) 公表特許公報（Ａ）　(11)特許出願公表番号
特表平10－512070
(43)公表日　平成10年(1998)11月17日

(51)Int.Cl.[6]	識別記号	ＦＩ	
Ｇ０６Ｆ　9/30	３５０	Ｇ０６Ｆ　9/30	３５０Ｅ

審査請求　未請求　予備審査請求　有　（全 56 頁）

(21)出願番号　特願平8－519125
(86)（22)出願日　平成７年(1995)12月１日
(85)翻訳文提出日　平成９年(1997)６月２日
(86)国際出願番号　ＰＣＴ／ＵＳ９５／１５７１３
(87)国際公開番号　ＷＯ９６／１７２９１
(87)国際公開日　平成８年(1996)６月６日
(31)優先権主張番号　０８／３４９，０４７
(32)優先日　1994年12月２日
(33)優先権主張国　米国（ＵＳ）

(71)出願人　インテル・コーポレーション
アメリカ合衆国　95052　カリフォルニア州・サンタ　クララ・ミッション　カレッジ　ブーレバード・2200
(72)発明者　ペレグ，アレキザンダー
イスラエル国・ハイファ・カメリア・ハンナ　ストリート，38
(72)発明者　ヤーリ，ヤーコブ
イスラエル国・ハイファ・ハナディン・ソーロ　ハナディン・17／2
(74)代理人　弁理士　山川　政樹　（外５名）

最終頁に続く

(54)【発明の名称】　複合オペランドのパック演算機能を有するマイクロプロセッサ

(57)【要約】
プロセッサが、第１のパックドデータを格納する第１のレジスタ（２０９）と、デコーダ（２０２）と、機能ユニット（２０３）とを備える。デコーダは、第１の制御信号と第２の制御信号を受け取る制御信号入力（２０７）を有する。第１の制御信号はパック演算を示し、第２の制御信号はアンパック演算を示す。機能ユニットはデコーダ（２０２）とレジスタ（２０９）に結合されている。機能ユニットは、第１のパックドデータを使用してパック演算とアンパック演算を行うほか、移動演算も行う。

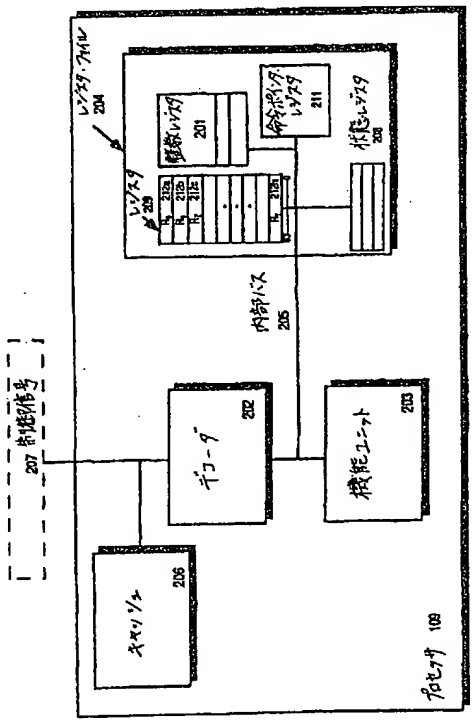


Figure 2

【特許請求の範囲】

１．第１のパックドデータを格納する第１のレジスタと、

パック演算を示す第１の制御信号とアンパック演算を示す第２の制御信号とを受け取る制御信号入力と、

前記デコーダと前記第１のレジスタとに結合され、前記第１のパックドデータを使用して前記パック演算と前記アンパック演算を実行する機能ユニットとを備えるプロセッサ。

２．前記第１のパックドデータが複数のデータ要素を含み、前記複数のデータ要素の各データ要素がサイズを有し、前記第１の制御信号が前記サイズに対応するインジケータをさらに含むことを特徴とする、請求項１に記載のプロセッサ。

３．前記サイズがパックドバイトとパックドワードとパックドダブルワードとのうちの１つであることを特徴とする、請求項２に記載のプロセッサ。

４．前記第１のパックドデータが６４ビットであることを特徴とする、請求項２に記載のプロセッサ。

５．前記第１の制御信号がソース・アドレスを含むことを特徴とする、請求項１に記載のプロセッサ。

６．前記第１の制御信号が符号インジケータを含み、前記符号インジケータが前記パック演算を符号付きと無符号のどちらで行うかを決定することを特徴とする、請求項１に記載のプロセッサ。

７．前記制御信号入力がさらに、移動演算を示す第３の制御信号を受け取ることができ、前記プロセッサが第２のレジスタをさらに備え、前記第２のレジスタに整数データが格納され、前記移動演算によって前記プロセッサが前記第１のレジスタと前記第２のレジスタの間でデータを移動することを特徴とする、請求項１に記載のプロセッサ。

８．前記プロセッサが、レジスタを含むレジスタ・ファイルをさらに備え、前記第１の制御信号と前記第２の制御信号が第２の場所を含み、前記第２の場所が前記レジスタに対応し、前記機能ユニットがさらに演算の結果を生成し、前記結果が前記第２の場所に格納されることを特徴とする、請求項１に記載のプロセッ

サ。

９．前記プロセッサの第１の制御信号が第２の場所を含み、前記第２の場所が記憶場所に対応し、前記機能ユニットがさらに演算の実行の結果を生成し、前記結果が前記第２の場所に格納されることを特徴とする、請求項１に記載のプロセッサ。

１０．デコーダと機能ユニットと第１のレジスタと第２のレジスタとを備え、前記デコーダが前記機能ユニットと前記第１のレジスタと前記第２のレジスタとに結合されたプロセッサにおいてパックドデータを操作する方法であって、

前記デコーダが制御信号をデコードするステップと、

前記制御信号がパック演算を示していると前記デコーダが判断した場合、前記プロセッサが、

前記第１のレジスタに格納されている第１のパックドデータにアクセスするステップと、

前記第２のレジスタに格納されている第２のパックドデータにアクセスするステップと、

前記第１のパックドデータを前記第２のパックドデータと共にパックして結果パックドデータを生成するステップと、

前記結果パックドデータを第１のレジスタに格納するステップとを実行するステップと、

前記制御信号がアンパック演算を示していると前記デコーダが判断した場合、前記プロセッサが、

前記第１のレジスタに格納されている前記第１のパックドデータにアクセスするステップと、

前記第２のレジスタに格納されている前記第２のパックドデータにアクセスするステップと、

前記第１のパックドデータと前記第２のパックドデータをアンパックして前記結果パックドデータを生成するステップと、

前記結果パックドデータを前記第１のレジスタに格納するステップ

とを実行するステップと、

前記制御信号が移動演算を示していると前記デコーダが判断した場合、前記プロセッサが、

前記第２のレジスタに格納されている前記第２のパックドデータにアクセスするステップと、

整数データを記憶する第３のレジスタに前記第２のパックドデータの一部を格納するステップと

を含む、プロセッサにおいてパックドデータを操作する方法。

１１．前記パックが飽和なしで行われ、前記パックが前記第１のパックドデータ内および前記第２のパックドデータ内の各データ要素の下位ビットにアクセスするステップを含み、前記下位ビットが前記結果パックドデータに含まれることを特徴とする、請求項１０に記載の方法。

１２．前記第１のパックドデータが第１の複数のデータ要素を含み、前記第１の複数のデータ要素の各データ要素が第１の所定のビット数で表され、前記制御信号がサイズインジケータを含み、前記サイズインジケータが前記第１の所定のビット数を示し、前記結果パックドデータが第２の複数のデータ要素を含み、前記第２の複数のデータ要素が前記第１の複数のデータ要素の２倍のデータ要素を有し、前記第２の複数のデータ要素の各データ要素が第２の所定のビット数で表され、前記第２の所定のビット数が前記第１のビット数の半分であることを特徴とする、請求項１０に記載の方法。

１３．前記第１のパックドデータが第１の組のデータ要素を含み、前記第２のパックドデータが第２の組のデータ要素を含み、前記アンパックが前記第１の組のデータ要素の下位データ要素を前記第２の組のデータ要素の下位データ要素によってインタリーブすることを特徴とする、請求項１０に記載の方法。

１４．前記第１のレジスタが６４ビット長であり、前記第１のパックドデータが４個のパックドワード・データ要素を含むことを特徴とする、請求項１０に記載の方法。

【発明の詳細な説明】

複合オペランドのパック演算機能を有するマイクロプロセッサ

発明の背景

発明の分野

本発明は、単一の制御信号を使用して複数のデータ要素を操作する演算を行う装置および方法を含む。本発明は、パックされたデータ（パックドデータ）・データ・タイプに対する移動演算、パック演算、およびアンパック演算の実行を可能にする。

関連技術の説明

現在、ほとんどのパーソナル・コンピュータ・システムは１つの命令によって演算を行って１つの結果を出す。命令の実行速度とプロセッサ命令の複雑さを増すことと、複数の命令を並列して実行することによって、パフォーマンスの向上が実現され、これは複雑命令セット・コンピュータ（ＣＩＳＣ，Complex Instruction Set Computer）と呼ばれる。米国カリフォルニア州サンタクララ所在のインテル・コーポレイションが販売するＩｎｔｅｌ８０２８６™マイクロプロセッサのようなプロセッサがＣＩＳＣプロセッサの範疇に入る。

以前のコンピュータ・システム・アーキテクチャは、ＣＩＳＣの概念を利用するように最適化されていた。そのようなシステムは一般に、３２ビット幅のデータ・バスを持つ。しかし、コンピュータ・サポーテッド・コオペレーション（ＣＳＣ：電子会議と混在メディア・データ操作を統合したもの）、二次元／三次元グラフィックス、画像処理、ビデオ圧縮／圧縮解除、認識アルゴリズム、および音声操作を対象にしたアプリケーションによって、パフォーマンス向上の必要が増す。しかし、命令の実行速度と複雑さを増すことが唯一の解決策である。

これらのアプリケーションの１つの一般的な点は、数ビットだけが重要な、大量のデータを操作することが多いことである。すなわち、意味のあるビットがデータ・バスのサイズよりもはるかに少ないビット数で表されるデータである。たとえば、プロセッサは８ビットおよび１６ビットのデータ（たとえばビデオ画像

における画素の色成分）に対して多くの演算を実行するが、それよりかなり広い

でデータ・バスとレジスタを有する。したがって、３２ビットのデータ・バスとレジスタを有し、これらのアルゴリズムの１つを実行するプロセッサは、データの先頭８ビットだけが重要であるため、そのデータの処理、伝送、および記憶容量の最大７５パーセントが無駄になることがある。

したがって、操作するデータを表すのに必要なビット数とプロセッサの実際のデータ伝送および記憶容量との差をより効率的に使用することによってパフォーマンスを向上させるプロセッサが望ましい。

発明の概要

複数のデータ要素に対して作用するシフト演算（シフトオペレーション）を有するプロセッサについて説明する。

このプロセッサは、第１のパックドデータを格納する第１のレジスタと、デコーダと、機能ユニットとを備える。デコーダは、制御信号入力を有する。制御信号入力は、第１の制御信号と第２の制御信号を受け取る。第１の制御信号はパック演算を示す。第２の制御信号はアンパック演算を示す。機能ユニットはデコーダとレジスタとに結合されている。機能ユニットは、第１のパックドデータを使用してパック演算とアンパック演算を行う。プロセッサは移動演算もサポートする。

本説明および図には多くの詳細が含まれるが、本発明は請求の範囲によって定義される。本発明には、それらの請求の範囲に記載されている限定だけが適用される。

図面の簡単な説明

本発明を、図面に限定的なものではなく例として図示する。同様の参照符号は同様の要素を示す。

第１図は、本発明の方法および装置を使用するコンピュータ・システムの実施例を示す図である。

第２図は、本発明のプロセッサの実施例を示す図である。

第３図は、プロセッサがレジスタ・ファイル内のデータを操作するために使用する一般的なステップを示す流れ図である。

第４ａ図は、記憶データ・タイプを示す図である。

第４ｂ図、第４ｃ図、および第４ｄ図は、レジスタ内整数データ表現を示す図である。

第５ａ図は、パックドデータ・タイプを示す図である。

第５ｂ図、第５ｃ図、および第５ｄ図は、レジスタ内パックドデータ表現を示す図である。

第６ａ図は、パックドデータの使用を示す、コンピュータ・システムで使用される制御信号形式の図である。

第６ｂ図は、パックドデータまたは整数データの使用を示す、コンピュータ・システムで使用可能な第２の制御信号形式の図である。

第７図は、パックドデータに対してシフト演算を行うときにプロセッサが従う方法の一実施例を示す図である。

第８ａ図は、パックドバイト・データに対するパック演算を実施することができる回路を示す図である。

第８ｂ図は、パックドワード・データに対するパック演算を実施することができる回路を示す図である。

第９図は、パックドデータに対するアンパック演算を行う場合にプロセッサがたどる方法の一実施例を示す図である。

第１０図は、パックドデータに対するアンパック演算を実施することができる回路を示す図である。

好ましい実施例の説明

本発明の一実施例の概要

複数のデータ要素に対して作用する移動演算（ムーブオペレーション）とパック演算とアンパック演算の機能を有するプロセッサについて説明する。以下の説明では、本発明を十分に理解することができるように、回路などの多くの特定の詳細を記載する。他の場合には、本発明が無用に不明瞭にならないように、周知の構造および技法については詳細に示さない。

定義

本発明の実施態様の説明を理解する基礎となるように、以下のように定義を示す。

ビットＸないしビットＹ：二進数のサブフィールドを規定する。たとえば、（基数２で示した）バイト００１１１０１０_2のビット６ないしビット０は、１サブフィールド１１０１０_2を表す。二進数の後の下付け文字「２」は基数２を示す。したがって、１０００_2は８_{10}に等しく、$Ｆ_{16}$は１５_{10}に等しい。

R_x：レジスタである。レジスタは、データの記憶と供給を行うことができる任意の素子である。レジスタの他の機能については後述する。レジスタはプロセッサのパッケージの一部であるとは限らない。

ＤＥＳＴ：データ・アドレスである。

ＳＲＣ１：データ・アドレスである。

ＳＲＣ２：データ・アドレスである。

結果(Result)：ＤＥＳＴによってアドレス指定されたレジスタに格納されるデータである。

ソース１：ＳＲＣ１によってアドレス指定されたレジスタに格納されているデータである。

ソース２：ＳＲＣ２によってアドレス指定されたレジスタに格納されているデータである。

コンピュータ・システム

第１図を参照すると、本発明の実施例を実施することができるコンピュータ・システムが、コンピュータ・システム１００として図示されている。コンピュータ・システム１００は、情報を伝送するバス１０１またはその他の通信ハードウェアおよびソフトウェアと、バス１０１に結合された情報を処理するプロセッサ１０９とを備える。コンピュータ・システム１００はさらに、バス１０１に結合され、プロセッサ１０９によって実行される情報と命令を記憶するランダム・アクセス・メモリ（ＲＡＭ）またはその他のダイナミック記憶装置（メイン・メモリ１０４と呼ぶ）を備える。メイン・メモリ１０４は、プロセッサ１０９による命令の実行中に変数またはその他の中間情報を一時的に記憶するためにも使用す

ることができる。コンピュータ・システム１００は、バス１０１に結合され、プロセッサ１０９のための静的情報および命令を記憶する読取り専用メモリ（ＲＯＭ）１０６またはその他のスタティック記憶装置あるいはその両方も備える。バス１０１には情報と命令を記憶するデータ記憶装置１０７が結合されている。

さらに、コンピュータ・システム１００には磁気ディスクや光ディスクなどのデータ記憶装置１０７とそれに対応するディスク・ドライブを結合することができる。コンピュータ・システム１００は、コンピュータ・ユーザに情報を表示するためにバス１０１を介して表示装置１２１にも結合することができる。表示装置１２１は、フレーム・バッファ、専用グラフィックス・レンダリング装置、陰極線管（ＣＲＴ）、フラット・パネル・ディスプレイを含むことができる。プロセッサ１０９に情報とコマンド選択を伝えるために、英数字およびその他のキーを備える英数字入力装置１２２が、典型的にはバス１０１に結合されている。他のタイプのユーザ入力装置は、プロセッサ１０９に指示情報とコマンド選択を伝え、表示装置１２１上のカーソル移動を制御する、マウス、トラックボール、ペン、タッチ画面、カーソル指示キーなどのカーソル制御装置１２３である。この入力装置は一般に、第１の軸（たとえばｘ）と第２の軸（たとえばｙ）の２つの軸に２つの自由度を持ち、それによってこの装置は平面内の位置を指定することができる。しかし、本発明は、２つの自由度しかもたない入力装置には限定されない。

バス１０１に結合することができる他の装置は、命令、データ、またはその他の情報を、紙、フィルム、または同様のタイプの媒体などの媒体に印刷するために使用することができるハード・コピー装置１２４である。さらに、コンピュータ・システム１００は、情報を記録するためにマイクロホンに結合されたオーディオ・ディジタイザなど、録音または再生あるいはその両方のための装置１２５に結合することができる。さらに、この装置は、ディジタル化された音声を再生するためにディジタル－アナログ（Ｄ／Ａ）変換器に結合されたスピーカも含むことができる。

また、コンピュータ・システム１００は、コンピュータ・ネットワーク（たと

えばＬＡＮ）内の端末とすることもできる。その場合、コンピュータ・システム１００は、いくつかのネットワーク化された装置を含むコンピュータ・システムのコンピュータ・サブシステムとなる。コンピュータ・システム１００は、任意選択としてビデオ・ディジタイジング装置１２６を備える。ビデオ・ディジタイジング装置１２６を使用して、ビデオ画像をキャプチャし、それをコンピュータ・ネットワーク上の他の装置に送信することができる。

コンピュータ・システム１００は、コンピュータ・サポーテッド・コオペレーション（ＣＳＣ：電子会議と混在媒体データ操作とが統合されたもの）、二次元／三次元グラフィックス、画像処理、ビデオ圧縮／圧縮解除、認識アルゴリズム、および音声操作に対応するのに有用である。

プロセッサ

第２図にプロセッサ１０９の詳細図を示す。プロセッサ１０９は、ＢｉＣＭＯＳ、ＣＭＯＳ、ＮＭＯＳなどのいくつかの処理技法のいずれかを使用して、１つまたは複数の基板上に実装することができる。

プロセッサ１０９は、プロセッサ１０９が使用する制御信号とデータをデコードするデコーダ２０２を備える。その場合、データは内部バス２０５を介してレジスタ・ファイル２０４に格納することができる。明確に言えば、実施例のレジスタは特定のタイプの回路にのみに限定されることを意味しない。むしろ実施例のレジスタは、データの記憶および供給と、本明細書に記載の機能を実行することができればよい。

データは、データのタイプに応じて、整数レジスタ２０１、レジスタ２０９、状態レジスタ２０８、または命令ポインタ・レジスタ２１１に格納することができる。たとえば浮動小数点レジスタなど他のレジスタをレジスタ・ファイル２０４に含めることができる。一実施例では、整数レジスタ２０１には、３２ビットの整数データが格納される。一実施例では、レジスタ２０９には$R_0$２１２ａ～$R_7$２１２ｈの８個のレジスタが含まれる。レジスタ２０９内の各レジスタ長は

６４ビットである。$R_1$２１２ａ、$R_2$２１２ｂ、および$R_3$２１２ｃがレジスタ２０９内の個々のレジスタの例である。レジスタ２０９内の３２ビットのレジス

タを、整数レジスタ２０１内の整数レジスタに移動させることができる。同様に、整数レジスタ内の値をレジスタ２０９内の３２ビットのレジスタに移動させることができる。

状態レジスタ２０８は、プロセッサ１０９の状況を示す。命令ポインタ・レジスタ２１１には、次に実行される命令のアドレスが格納される。整数レジスタ２０１，レジスタ２０９、状態レジスタ２０８、および命令ポインタ・レジスタ２１１はすべて内部バス２０５に接続されている。内部バスには任意の追加のレジスタが接続される。

他の実施例では、これらのレジスタのうちのいくつかは２つの異なるタイプのデータに使用することができる。たとえば、レジスタ２０９と整数レジスタ２０１を組み合わせて、各レジスタに整数データまたはパックドデータを格納することができる。他の実施例では、レジスタ２０９を浮動小数点レジスタとして使用することができる。この実施例では、パックドデータはレジスタ２０９または浮動小数点データに格納することができる。一実施例では、組み合わされたレジスタの長さは６４ビットで、整数は６４ビットで表される。この実施例では、パックドデータと整数データを格納する際に、レジスタはその２つのデータ・タイプを区別する必要がない。

機能ユニット２０３は、プロセッサ１０９が行う演算（オペレーション）を実行する。このような演算には、シフト、加算、減算、乗算などが含まれる。機能ユニット２０３は内部バス２０５に接続している。キャッシュ２０６は、プロセッサ１０９の任意選択要素であり、たとえばメイン・メモリ１０４からのデータまたは制御信号あるいはその両方をキャッシュするために使用される。キャッシュ２０６は、デコーダ２０２に接続され、制御信号２０７を受信するように接続されている。

第３図に、プロセッサ１０９の動作概要を示す。すなわち、第３図にはプロセッサ１０９がパックドデータに対する演算、アンパックドデータに対する演算、または他の何らかの操作を実行する間にたどるステップが示されている。たとえば、このような操作にはレジスタ・ファイル２０４内のレジスタに、キャッシュ

２０６、メイン・メモリ１０４、読取り専用メモリ（ＲＯＭ）１０６、またはデータ記憶装置１０７からデータをロードする操作が含まれる。本発明の一実施例では、プロセッサ１０９は、米国カリフォルニア州サンタクララ所在のインテル・コーポレイションが販売するＩｎｔｅｌ８０４８６™によってサポートされる命令のほとんどをサポートする。本発明の他の実施例では、プロセッサ１０９は米国カリフォルニア州サンタクララ所在のインテル・コーポレイションが販売するＩｎｔｅｌ８０４８６™によってサポートされるすべての演算をサポートする。本発明の他の実施例では、プロセッサ１０９は、すべて米国カリフォルニア州サンタクララ所在のインテル・コーポレイションが販売するＰｅｎｔｉｕｍ™プロセッサ、Ｉｎｔｅｌ８０４８６™プロセッサ、８０３８６™プロセッサ、Ｉｎｔｅｌ８０２８６™プロセッサ、およびＩｎｔｅｌ８０８６™プロセッサによってサポートされるすべての演算をサポートする。本発明の他の実施例では、プロセッサ１０９は、米国カリフォルニア州サンタクララ所在のインテル・コーポレイションが定義するＩＡ™（インテル・アーキテクチャ）でサポートされるすべての演算をサポートする（米国カリフォルニア州サンタクララのインテルから入手可能な「Ｍｉｃｒｏｐｒｏｃｅｓｓｏｒｓ，Ｉｎｔｅｌ　Ｄａｔａ　Ｂｏｏｋｓ　ｖｏｌｕｍｅ１およびｖｏｌｕｍｅ２、１９９２年および１９９３年刊」を参照）。一般に、プロセッサ１０９はＰｅｎｔｉｕｍ™プロセッサの現行命令セットをサポートすることができるが、将来の命令と本明細書に記載の命令を組み込むように修正することもできる。重要なのは、汎用プロセッサ１０９が、本明細書に記載の演算に加えて、従来使用されていた演算をサポートすることができることである。

ステップ３０１で、デコーダ２０２がキャッシュ２０６またはバス１０１から制御信号２０７を受け取る。デコーダ２０２は、制御信号をデコードして、実行すべき演算を判断する。

ステップ３０２で、デコーダ２０２はレジスタ・ファイル２０４またはメモリ内の記憶場所にアクセスする。制御信号２０７で指定されたレジスタ・アドレスに応じて、レジスタ・ファイル２０４内のレジスタかメモリ内の記憶場所のどち

らかにアクセスする。たとえば、パックドデータに対する演算の場合、制御信号２０７にはＳＲＣ１、ＳＲＣ２、およびＤＥＳＴレジスタ・アドレスを含めることができる。ＳＲＣ１は第１のソース・レジスタのアドレスである。ＳＲＣ２は第２のソース・レジスタのアドレスである。すべての演算が２つのソース・アドレスを必要とするわけではないので、場合によってはＳＲＣ２アドレスは任意選択である。ＳＲＣ２アドレスが不要な場合、ＳＲＣ１アドレスのみが使用される。ＤＥＳＴは、結果データが格納される宛先レジスタのアドレスである。一実施例では、ＳＲＣ１またはＳＲＣ２はＤＥＳＴとしても使用される。ＳＲＣ１、ＳＲＣ２、およびＤＥＳＴについては第６ａ図および第６ｂ図を参照しながら詳述する。対応するレジスタに格納されているデータをそれぞれＳｏｕｒｃｅ１、Ｓｏｕｒｃｅ２、およびＲｅｓｕｌｔと呼ぶ。これらの各データの長さは６４ビットである。

本発明の他の実施例では、ＳＲＣＩ、ＳＲＣ２、およびＤＥＳＴのいずれか１つまたは全部は、プロセッサ１０９のアドレス可能記憶空間内の記憶場所を規定することができる。たとえば、ＳＲＣ１はメイン・メモリ１０４内の記憶場所を識別し、ＳＲＣ２は整数レジスタ２０１内の第１のレジスタを識別し、ＤＥＳＴはレジスタ２０９内の第２のレジスタを識別する。本明細書では説明を簡単にするために、レジスタ・ファイル２０４へのアクセスについて言及するが、これらのアクセスはレジスタ・ファイル２０４の代わりにメモリに対して行うこともできる。

本発明の他の実施例では、命令コードはＳＲＣ１とＳＲＣ２の２つのアドレスしか含まない。この実施例では、演算の結果はＳＲＣ１レジスタまたはＳＲＣ２レジスタに格納される。すなわち、ＳＲＣ１（またはＳＲＣ２）をＤＥＳＴとして使用する。このタイプのアドレス指定は、２つのアドレスしか持たない以前のＣＩＳＣ命令に対応する。これによって、デコーダ２０２における複雑さが減少する。この実施例では、ＳＲＣ１レジスタに含まれるデータを破壊してはならない場合、演算を実行する前にそのデータをまず別のレジスタにコピーしなければならないことに留意されたい。コピーには追加の命令が必要になる。本明細書では説明を簡単にするために、３アドレスのアドレス指定方式について説明する

（すなわちＳＲＣ１、ＳＲＣ２、およびＤＥＳＴ）。しかし、一実施例では制御信号にＳＲＣ１とＳＲＣ２しか含めることができず、ＳＲＣ１（またはＳＲＣ２）によって宛先レジスタを識別することを想起されたい。

制御信号が演算を要求とする場合、ステップ３０３で、機能ユニット２０３がレジスタ・ファイル２０４内のアクセス・データに対してその演算を実行するように使用可能にされる。機能ユニット２０３で演算が実行されると、ステップ３０４でその結果が制御信号２０７の要件に従ってレジスタ・ファイル２０４に戻されて格納される。

データ形式および記憶形式

第４ａ図に、第１図のコンピュータ・システムで使用可能なデータ形式をいくつか示す。これらのデータ形式は固定小数点である。プロセッサ１０９はこれらのデータ形式を操作することができる。マルチメディア・アルゴリズムはこれらのデータ形式を使用することが多い。バイト４０１は８ビットの情報を含む。ワード４０２は１６ビットの情報、すなわち２バイトを含む。ダブルワード４０３は３２ビットの情報、すなわち４バイトを含む。したがって、プロセッサ１０９はこれらの記憶データ形式のうちの任意の１つに対して操作を行うことができる制御信号を実行する。

以下の説明では、ビット、バイト、ワード、およびダブルワード・サブフィールドについて言及する。たとえば、（基数２で示す）バイト00111010_2のビット６ないしビット０はサブフィールド111010_2を表す。

第４ｂ図ないし第４ｄ図に、本発明の一実施例で使用するレジスタ内表現を示す。たとえば、無符号バイトのレジスタ内表現４１０によって、整数レジスタ２０１内のレジスタに格納されているデータを表すことができる。一実施例では、整数レジスタ２０１内のレジスタ長は６４ビットである。他の実施例では、整数レジスタ２０１内のレジスタ長は３２ビットである。説明を簡単にするために、以下の説明では６４ビットの整数レジスタについて説明するが、３２ビットの整数レジスタを使用することもできる。

無符号バイトのレジスタ内表現４１０は、プロセッサ１０９が整数レジスタ２

０１にバイト４０１を格納し、そのレジスタ内のビット７ないしビット０の先頭８ビットがそのデータ・バイト４０１専用であることを示している。これらのビットを｛ｂ｝と示す。このバイトを正しく表すには、残りの５６ビットがゼロでなければならない。符号付きバイトのレジスタ内表現４１１の場合、整数レジスタ２０１にはデータはビット６ないしビット０の先頭７ビットにデータとして格納される。７番目のビットは符号ビットを表し、｛ｓ｝で示す。残りのビット６３ないしビット８はそのバイトの符号の継続である。

無符号ワードのレジスタ内表現４１２は、レジスタ２０１のうちの１つのレジスタに格納される。ビット１５ないしビット０には、無符号ワード４０２が入れられる。これらのビットを｛ｗ｝で示す。このワードを正しく表すには、残りのビット６３ないしビット１６はゼロでなければならない。符号付きワード４０２は、符号付きワードのレジスタ内表現４１３が示すように、ビット１４ないしビット０に格納される。残りのビット６３ないしビット１５は符号フィールドである。

ダブルワード４０３は、無符号ダブルワードのレジスタ内表現４１４または符号付きダブルワードのレジスタ内表現４１５として格納することができる。無符号ダブルワードのレジスタ内表現４１４のビット３１ないしビット０がデータである。これらのビットを｛ｄ｝で示す。この無符号ダブルワードを正しく表すには、残りのビット６３ないしビット３２はゼロでなければならない。整数レジスタ２０１には、符号付きダブルワードのレジスタ内表現４１５が、そのビット３０ないしビット０に格納される。残りのビット６３ないしビット３１は符号フィールドである。

前述の第４ｂ図ないし第４ｄ図に示すように、データ・タイプによっては６４ビット幅のレジスタに格納するのは非効率的な格納方法である。たとえば、無符号バイトのレジスタ内表現４１０を格納する場合、ビット６３ないしビット８はゼロでなければならず、ビット７ないしビット０にしか非ゼロ・ビットを入れることができない。したがって、６４ビット・レジスタに１バイトを格納するプロセッサは、レジスタの容量の１２．５％しか使用しない。同様に、機能ユニット２０３によって実行される命令は先頭の数ビットしか重要ではない。

第５ａ図に、パックされたデータすなわちパックドデータのデータ形式を示す。各パックドデータは複数の独立したデータ要素を含む。パックドバイト（パックされたバイト）５０１、パックドワード（パックされたワード）５０２、パックドダブルワード（パックされたダブルワード）５０３の３つのパックドデータ形式が図示されている。パックドバイトは、本発明の一実施例では６４ビット長であり、８個のデータ要素を含む。各データ要素は１バイト長である。一般に、データ要素は１つのレジスタ（または記憶場所）に同じ長さの他のデータ要素と共に格納される個々のデータである。本発明の一実施例では、１つのレジスタに格納されるデータ要素の数は、６４ビットをデータ要素のビット長で割った商である。

パックドワード５０２は６４ビット長であり、４個のワード４０２データ要素を含む。各ワード４０２データ要素は、１６ビットの情報を含む。

パックドダブルワード５０３は６４ビット長であり、２個のダブルワード４０３データ要素を含む。各ダブルワード４０３データ要素は３２ビットの情報を含む。

第５ｂ図ないし第５ｄ図にレジスタ内パックドデータ記憶表現を示す。無符号パックドバイトのレジスタ内表現５１０は、レジスタ$R_0$２１２ａ～R_n２１２ａｆのうちの１つにパックドバイト５０１が格納されている様子を示している。各バイト・データ要素の情報は、バイト０はビット７ないしビット０に格納され、バイト１はビット１５ないしビット８、バイト２はビット２３ないしビット１６、バイト３はビット３１ないしビット２４，バイト４はビット３９ないしビット３２、バイト５はビット４７ないしビット４０、バイト６はビット５５ないしビット４８、バイト７はビット６３ないしビット５６に格納される。したがって、レジスタ内ですべての使用可能ビットが使用される。この記憶構成によって、プロセッサの記憶効率が向上する。また、８個のデータ要素にアクセスして、１つの操作を８個のデータ要素に同時に実行することができるようになる。符号付きパックドバイトのレジスタ内表現５１１も同様にレジスタ２０９内のレジスタに格納される。どのバイト・データ要素でも８番目のビットのみが必要な符号ビットであり、他のビットは使用してもしなくても符号が示されることに留意された

い。

無符号パックドワードのレジスタ内表現５１２は、ワード３ないしワード０がレジスタ２０９のうちの１つのレジスタに格納される様子を示している。ビット１５ないしビット０にはワード０のデータ要素情報が入り、ビット３１ないしビット１６にはデータ要素ワード１の情報が入り、ビット４７ないしビット３２にはデータ要素ワード２の情報が入り、ビット６３ないしビット４８にはデータ要素ワード３の情報が入る。符号付きパックドワードのレジスタ内表現５１３は無符号パックドワードのレジスタ内表現５１２と同様である。各ワード・データ要素の１６番目のビットにのみ、必要な符号インジケータが入ることに留意されたい。

無符号パックドダブルワードのレジスタ内表現５１４は、レジスタ２０９に２個のダブルワード・データ要素が格納される様子を示している。ダブルワード０はレジスタのビット３１ないしビット０に格納される。ダブルワード１はレジスタのビット６３ないしビット３２に格納される。符号付きパックドダブルワードのレジスタ内表現５１５は無符号パックドダブルワードのレジスタ内表現５１４と同様である。必要な符号ビットはダブルワード・データ要素の３２番目のビットであることに留意されたい。

前述のように、レジスタ２０９はパックドデータと整数データの両方に使用することができる。本発明のこの実施例では、アドレス指定されたレジスタ、たとえばR$_1$２１２ａにパックドデータと単純整数／固定小数点データのどちらが格納されているかを追跡するために、個々のプログラミング・プロセッサ１０９が必要である。他の実施例では、プロセッサ１０９はレジスタ２０９の個々のレジスタに格納されているデータのタイプを追跡することができる。この代替実施例では、たとえば単純／固定小数点整数データに対してパック加算を行おうとした場合、エラーを生成することができる。

制御信号の形式

以下に、プロセッサ１０９がパックドデータを操作するために使用する制御信号形式の一実施例について説明する。本発明の一実施例では、制御信号は３２ビ

ットで表される。デコーダ２０２はバス１０１から制御信号２０７を受け取ることができる。他の実施例では、デコーダ２０２はキャッシュ２０６からもそのような制御信号を受け取ることができる。

第６ａ図にパックドデータを操作する制御信号の一般的な形式を示す。命令フィールドＯＰ６０１（ビット３１ないしビット２６）は、たとえば、パック加算、パック減算など、プロセッサ１０９によって実行される演算に関する情報を与える。ＳＲＣ１　６０２（ビット２５ないしビット２０）は、レジスタ２０９内のレジスタのソース・レジスタ・アドレスを供給する。このソース・レジスタは、制御信号の実行で使用される第１のパックドデータＳｏｕｒｃｅ１を保持する。同様に、ＳＲＣ２　６０３（ビット１９ないしビット１４）には、レジスタ２０９内のレジスタのアドレスが入れられる。この第２のソース・レジスタは、演算の実行時に使用されるパックドデータＳｏｕｒｃｅ２を保持する。ＤＥＳＴ６０５（ビット５ないしビット０）にはレジスタ２０９内のレジスタのアドレスが入れられる。この宛先レジスタには、パックドデータ演算の結果パックドデータＲｅｓｕｌｔが格納される。

制御ビットＳＺ６１０（ビット１２およびビット１３）は、第１および第２のパックドデータ・ソース・レジスタ内のデータ要素の長さを示す。ＳＺ６１０が01_2に等しい場合、パックドデータはパックドバイト５０１としてフォーマットされる。ＳＺ６１０が10_2に等しい場合、パックドデータはパックドワード５０２としてフォーマットされる。しかし、00_2または11_2と等しいＳＺ６１０を受け取った場合、他の実施例では、これらの値のうちの１つを使用してパックドダブルワード５０３を示すことができる。

制御ビットＴ６１１（ビット１１）は、演算を飽和モードで行うかどうかを示す。Ｔ６１１が１の場合、飽和演算が行われる。Ｔ６１１がゼロの場合、非飽和演算が行われる。飽和演算については後述する。

制御ビットＳ６１２（ビット１０）は、符号付き演算の使用を示す。Ｓ６１２が１の場合、符号付き演算が行われる。Ｓ６１２がゼロの場合、無符号演算が行われる。

第６ｂ図に、パックドデータを操作する制御信号の第２の一般的形式を示す。この形式は、米国イリノイ州マウント・プロスペクトＰ．Ｏ．Ｂｏｘ７６４１インテル・コーポレイションのＬｉｔｅｒａｔｕｒｅ　Ｓａｌｅｓから入手可能な”Ｐｅｎｔｉｕｍ™ Ｐｒｏｃｅｓｓｏｒ　Ｆａｍｉｌｙ　Ｕｓｅｒ’ｓ　Ｍａｎｕａｌ”に記載されている汎用整数命令コード形式に対応する。ＯＰ６０１、ＳＺ６１０、Ｔ６１１、およびＳ６１２がすべて組み合わされて１つの大きなフィールドになることに留意されたい。制御信号によっては、ビット３ないし５がＳＲＣ１　６０２となる。一実施例では、ＳＲＣ１　６０２アドレスがある場合、ビット３ないし５はＤＥＳＴ６０５にも対応する。ＳＲＣ２　６０３アドレスが存在する代替実施例ではビット０ないし２もＤＥＳＴ６０５に対応する。パックドシフト即値演算のような他の制御信号の場合、ビット３ないし５は命令コード・フィールドの拡張部を表す。一実施例では、この拡張部によってプログラマはシフト・カウント値などの即値を制御信号と共に組み込むことができる。一実施例では、即値は制御信号の後に続く。これについては”Ｐｅｎｔｉｕｍ™ Ｐｒｏｃｅｓｓｏｒ　Ｆａｍｉｌｙ　Ｕｓｅｒ’ｓ　Ｍａｎｕａｌ”の付録ＦのＦ－１～Ｆ－３ページに詳述されている。ビット０ないし２はＳＲＣ２　６０３を表す。この汎用形式によって、レジスタからレジスタ、メモリからレジスタ、メモリによるレジスタ、レジスタによるレジスタ、即値によるレジスタ、レジスタからメモリのアドレス指定を行うことができる。また、一実施例では、この汎用形式は整数レジスタからレジスタと、レジスタから整数レジスタへのアドレス指定もサポートする。

飽和／非飽和の説明

前述のように、Ｔ６１１は演算が任意選択で飽和するかどうかを示す。飽和を可能にした演算の結果がデータの範囲からオーバーフローまたはアンダーフローする場合、その結果はクランプされる。クランプとは、結果がその範囲の最大値または最小値を超える場合、その結果を最大値または最小値に設定することを意味する。アンダーフローの場合、飽和によって結果がその範囲内の最低値にクランプされ、オーバーフローの場合は最高値にクランプされる。各データ形式の許

容範囲を表１に示す。

表１

データ形式	最小値	最大値
無符号バイト	0	２５５
符号付きバイト	－１２８	１２７
無符号ワード	0	６５５３５
符号付きワード	－３２７６８	３２７６７
無符号ダブルワード	0	$2^{64}-1$
符号付きダブルワード	-2^{63}	$2^{63}-1$

前述のように、Ｔ６１１は飽和演算を行うかどうかを示す。したがって、無符号バイト・データ形式を使用し、演算結果＝２５８で、飽和を使用可能にしていた場合、結果は演算の宛先レジスタに格納される前に２５５にクランプされることになる。同様に、演算結果＝－３２９９９で、プロセッサ１０９が飽和を使用可能にして符号付きデータ形式を使用した場合、結果は演算の宛先レジスタに格納される前に－３２７６８にクランプされることになる。

データ操作演算

本発明の一実施例では、標準ＣＩＳＣ命令セット（アンパックドデータ演算）をサポートするだけでなくパックドデータのシフト演算もサポートすることによって、マルチメディア・アプリケーションのパフォーマンスを向上させる。このようなパックドデータ演算には、加算、減算、乗算、比較、シフト、ＡＮＤ、およびＸＯＲを含めることができる。しかし、これらの演算を十分に利用するためには、データ操作演算を組み込む必要があると判断されている。このようなデータ操作演算には、移動、パック、アンパックを含めることができる。移動、パック、およびアンパックは、プログラマにとってより使いやすい形式のパックドデ

ータを生成することによって他の演算の実行を容易にする。

他のパック演算の詳細な背景については、　　年　月　日出願の特許出願第　　号“Ａ Ｍｉｃｒｏｐｒｏｃｅｓｓｏｒ Ｈａｖｉｎｇ ａ Ｃｏｍｐａｒｅ Ｏｐｅｒａｔｉｏｎ”、　　年　月　日出願の特許出願第　　号“Ａ Ｍｉｃｒｏｐｒｏｃｅｓｓｏｒ Ｈａｖｉｎｇ ａ Ｍｕｌｔｉｐｌｙ Ｏｐｅｒａｔｉｏｎ”、　　年　月　日出願の特許出願第　　号“Ａ Ｎｏｖｅｌ Ｐｒｏｃｅｓｓｏｒ Ｈａｖｉｎｇ Ｓｈｉｆｔ Ｏｐｅｒａｔｉｏｎｓ”、１９９３年１２月３０日出願の特許出願第０８／１７６１２３号”Ａ Ｍｅｔｈｏｄ ａｎｄ Ａｐｐａｒａｔｕｓ Ｕｓｉｎｇ Ｐａｃｋｅｄ Ｄａｔａ ｉｎ ａ Ｐｒｏｃｅｓｓｏｒ”、および１９９３年１２月３０日出願の特許出願第０８／１７５７７２号“Ａ Ｍｅｔｈｏｄ ａｎｄ Ａｐｐａｒａｔｕｓ Ｕｓｉｎｇ Ｎｏｖｅｌ Ｏｐｅｒａｔｉｏｎｓ ｉｎ ａ Ｐｒｏｃｅｓｓｏｒ”を参照されたい。これらはすべて本発明の譲渡人に譲渡される。

移動演算

移動演算は、レジスタ２０９との間でデータを送受信する。一実施例では、ＳＲＣ２　６０３がソース・データを含むアドレスであり、ＤＥＳＴ６０５がデータの送り先のアドレスである。この実施例では、ＳＲＣ１　６０２は使用しない。他の実施例では、ＳＲＣ１　６０２はＤＥＳＴ６０５である。

移動演算の説明のために、レジスタと記憶場所との区別を設ける。レジスタはレジスタ・ファイル２０４内にあり、記憶場所はたとえばキャッシュ２０６，メイン・メモリ１０４、ＲＯＭ１０６、データ記憶装置１０７内とすることができる。

移動演算によって、データを記憶場所からレジスタ２０９、レジスタ２０９から記憶場所、レジスタ２０９内のレジスタからレジスタ２０９内の第２のレジスタに移動することができる。一実施例では、パックドデータは整数データを記憶するために使用されるレジスタとは異なるレジスタに格納される。この実施例では、移動演算によって整数レジスタ２０１からレジスタ２０９にデータを移動す

ることができる。たとえば、プロセッサ１０９において、レジスタ２０９にパックドデータが格納され、整数レジスタ２０１に整数データが格納されている場合、移動命令を使用して整数レジスタ２０１からレジスタ２０９にデータを移動したり、その逆を行ったりすることができる。

一実施例では、移動のためにメモリ・アドレスを示すと、記憶場所（最下位バイトを示す記憶場所）にある８バイトのデータがレジスタ２０９内のレジスタにロードされるかまたはそのレジスタから格納される。レジスタ２０９内のレジスタが示された場合、そのレジスタの内容がレジスタ２０９内の第２のレジスタに移動されるかまたはそこからロードされる。整数レジスタ２０１が６４ビット長であって、整数レジスタを指定した場合、そのレジスタ内の８バイトのデータがレジスタ２０９内のレジスタにロードされるかまたはそのレジスタから格納される。

一実施例では、整数は３２ビットで表される。レジスタ２０９から整数レジスタ２０１への移動演算を行う場合、パックドデータの下位３２ビットだけが指定された整数レジスタに移動される。一実施例では、上位３２ビットはゼロにされる。同様に、整数レジスタ２０１からレジスタ２０９への移動を実行すると、レジスタ２０９内のレジスタの下位３２ビットだけがロードされる。一実施例では、プロセッサ１０９はレジスタ２０９内のレジスタとメモリとの間の３２ビットの移動演算をサポートする。他の実施例では、パックドデータの上位３２ビットに対して３２ビットのみの移動が行われる。

パック演算

本発明の一実施例では、ＳＲＣ１　６０２レジスタにデータ（Ｓｏｕｒｃｅ１）が入れられ、ＳＲＣ２　６０３レジスタにデータ（Ｓｏｕｒｃｅ２）が入れられ、ＤＥＳＴ６０５レジスタには演算の結果データ（Ｒｅｓｕｌｔ）が入れられる。すなわち、Ｓｏｕｒｃｅ１の部分とＳｏｕｒｃｅ２の部分が一緒にパックされて結果が生成される。

一実施例では、パック演算は、ソース・パックドワード（またはダブルワード）の下位バイト（またはワード）をＲｅｓｕｌｔのバイト（またはワード）にパッ

クすることによって、パックドワード（またはダブルワード）がパックドバイト（またはワード）に変換する。一実施例では、パック演算によって、クワッド・パックドワードがパックドダブルワードに変換される。この演算は、任意選択により符号付きデータを使用して行うことができる。さらにこの演算は、任意選択により飽和を使用して行うことができる。

第７図に、パックドデータに対してパック演算を行う方法の一実施例を示す。この実施例は、第２図のプロセッサ１０９で実施することができる。

ステップ７０１で、プロセッサ１０９が受け取った制御信号２０７をデコーダ２０２がデコードする。したがって、デコーダ２０２は、適切なシフト演算の命令コードと、レジスタ２０９内のＳＲＣ１　６０２、ＳＲＣ２　６０３、およびＤＥＳＴ６０５アドレスと、飽和／非飽和と、符号付き／無符号と、パックドデータ内のデータ要素の長さとをデコードする。前述のように、ＳＲＣ１　６０２（またはＳＲＣ２　６０３）をＤＥＳＴ６０５として使用することができる。

ステップ７０２で、ＳＲＣ１　６０２アドレスとＳＲＣ２　６０３アドレスが与えられた場合、内部バス２０５を介してデコーダ２０２がレジスタ・ファイル２０４内のレジスタ２０９にアクセスする。レジスタ２０９は機能ユニット２０３にＳＲＣ１　６０２レジスタに格納されているパックドデータ（Ｓｏｕｒｃｅ１）と、ＳＲＣ２　６０３レジスタに格納されているパックドデータ（Ｓｏｕｒｃｅ２）を供給する。すなわち、レジスタ２０９は、パックドデータを内部バス２０５を介して機能ユニット２０３に伝達する。

ステップ７０３で、デコーダ２０２は機能ユニット２０３が適切なパック演算を実行することができるようにする。デコーダ２０２は、さらに、内部バス２０５を介して飽和とＳｏｕｒｃｅ１内およびＳｏｕｒｃｅ２内のデータ要素のサイズも伝達する。任意選択で飽和を使用して、結果データ要素内のデータの値を最大化する。Ｓｏｕｒｃｅ１またはＳｏｕｒｃｅ２内のデータ要素の値が、Ｒｅｓｕｌｔのデータ要素が表すことができる値の範囲より大きいかまたは小さい場合、それに対応する結果データ要素がその最高値または最低値に設定される。たとえば、Ｓｏｕｒｃｅ１およびＳｏｕｒｃｅ２のワード・データ要素内の符号付き値が０×８０（またはダブルワードの場合は０×８０００）よりも小さい場合、

結

果のバイト（またはワード）データ要素が０×８０（またはダブルワードの場合は０×８０００）にクランプされる。Ｓｏｕｒｃｅ１およびＳｏｕｒｃｅ２のワード・データ要素内の符号付き値が０×７Ｆ（またはダブルワードの場合は０×７ＦＦＦ）よりも大きい場合は、結果のバイト（またはワード）データ要素が０×７Ｆ（または０×７ＦＦＦ）にクランプされる。

ステップ７１０で、データ要素のサイズによって次にどのステップを実行するかが決まる。データ要素のサイズが１６ビット（パックドワード５０２データ）の場合、機能ユニット２０３はステップ７１２を実行する。しかしパックドデータ内のデータ要素のサイズが３２ビット（パックドダブルワード５０３データ）の場合、機能ユニット２０３はステップ７１４を実行する。

ソース・データ要素のサイズが１６ビットであると仮定すると、ステップ７１２が実行される。ステップ７１２では、以下のように行う。Ｓｏｕｒｃｅ１のビット７～０はＲｅｓｕｌｔのビット７～０である。Ｓｏｕｒｃｅ１のビット２３～１６はＲｅｓｕｌｔのビット１５～８である。Ｓｏｕｒｃｅ１のビット３９～３２はＲｅｓｕｌｔのビット２３～１６である。Ｓｏｕｒｃｅ１のビット６３～５６はＲｅｓｕｌｔのビット３１～２４である。Ｓｏｕｒｃｅ２のビット７～０はＲｅｓｕｌｔのビット３９～３２である。Ｓｏｕｒｃｅ２のビット２３～１６はＲｅｓｕｌｔのビット４７～４０である。Ｓｏｕｒｃｅ２のビット３９～３２はＲｅｓｕｌｔのビット５５～４８である。Ｓｏｕｒｃｅ２のビット６３～５６はＲｅｓｕｌｔのビット３１～２４である。飽和を設定した場合は、Ｒｅｓｕｌｔデータ要素をクランプすべきかどうかを判断するために各ワードの上位ビットが検査される。

ソース・データ要素のサイズが３２ビットであると仮定すると、ステップ７１４が実行される。ステップ７１４では、以下のように行われる。Ｓｏｕｒｃｅ１のビット１５～０はＲｅｓｕｌｔのビット１５～０である。Ｓｏｕｒｃｅ１のビット４７～３２はＲｅｓｕｌｔのビット３１～１６である。Ｓｏｕｒｃｅ２のビット１５～０はＲｅｓｕｌｔのビット４７～３２である。Ｓｏｕｒｃｅ２のビッ

ト４７～３２はＲｅｓｕｌｔのビット６３～４８である。飽和を設定した場合は、Ｒｅｓｕｌｔデータ要素をクランプすべきかどうかを判断するために各ダブルワ

ードの上位ビットが検査される。

一実施例では、ステップ７１２のパック演算が同時に行われる。しかし、他の実施例では、このパック演算は順次に行われる。他の実施例では、このパック演算の一部が同時に行われ、一部は順次に行われる。これは、ステップ７１４のパック演算にも適用される。

ステップ７２０で、ＲｅｓｕｌｔがＤＥＳＴ６０５レジスタに格納される。

表２に、飽和を使用しないパック無符号ワード演算のレジスタ内表現を示す。最初の行のビットはＳｏｕｒｃｅ１のパックドデータ表現である。２番目の行のビットはＳｏｕｒｃｅ２のデータ表現である。３番目の行のビットはＲｅｓｕｌｔのパックドデータ表現である。各データ要素ビットの下の数字はデータ要素番号である。たとえば、Ｓｏｕｒｃｅ１データ要素３は10000000_2である。

表２

Source1

00101010 01010101	01010101 11111111	10000000 01110000	10001111 10001000
3	2	1	0

Source2

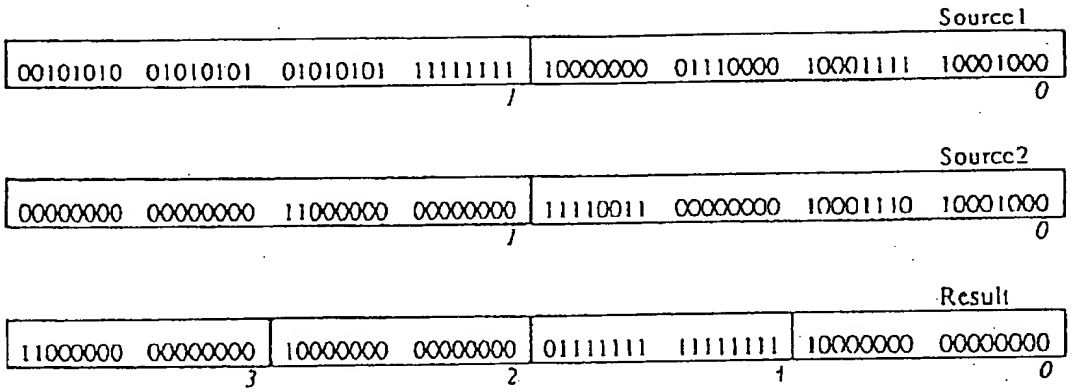
00000000 00000000	11000000 00000000	11110011 00000000	10001110 10001000
3	2	1	0

Result

00000000	00000000	00000000	10001000	01010101	11111111	01110000	10001000
7	6	5	4	3	2	1	0

表３に、飽和を使用したパック符号ダブルワード演算のレジスタ内表現を示す。

表３



パック回路

本発明の一実施例では、パック演算の効率的な実行を実現するために並列処理を使用する。第８ａ図および第８ｂ図に、パックドデータに対するパック演算を行う回路の一実施例を示す。

第８ａ図および第８ｂ図の回路は、演算制御回路８００と、結果レジスタ８５２と、結果レジスタ８５３と、８個の１６ビットから８ビットの飽和検査回路と、４個の３２ビットから１６ビットの飽和検査回路を備える。

演算制御回路８００は、デコーダ２０２から情報を受け取ってパック演算を可能にする。演算制御回路８００は飽和値を使用して各飽和検査回路の飽和検査を可能にする。ソース・パックドデータのサイズがワード・パックドデータ５０３の場合、演算制御回路８００によって出力イネーブル８３１が設定される。これによって出力レジスタ８５２の出力がイネーブルにされる。ソース・パックドデータのサイズがダブルワード・パックドデータ５０４の場合、演算制御回路８００によって出力イネーブル８３２が設定される。これによって、出力レジスタ８５３の出力がイネーブルにされる。

各飽和検査回路は飽和を選択的に検査することができる。飽和の検査を使用不能にした場合、各飽和検査回路はただ下位ビットを通過させて結果レジスタ内の対応する位置に入れるだけである。飽和の検査を使用可能にした場合、各飽和検査回路は上位ビットを検査して結果をクランプすべきかどうかを判断する。

飽和検査回路８１０ないし８１７は１６ビットの入力と８ビットの出力を有す

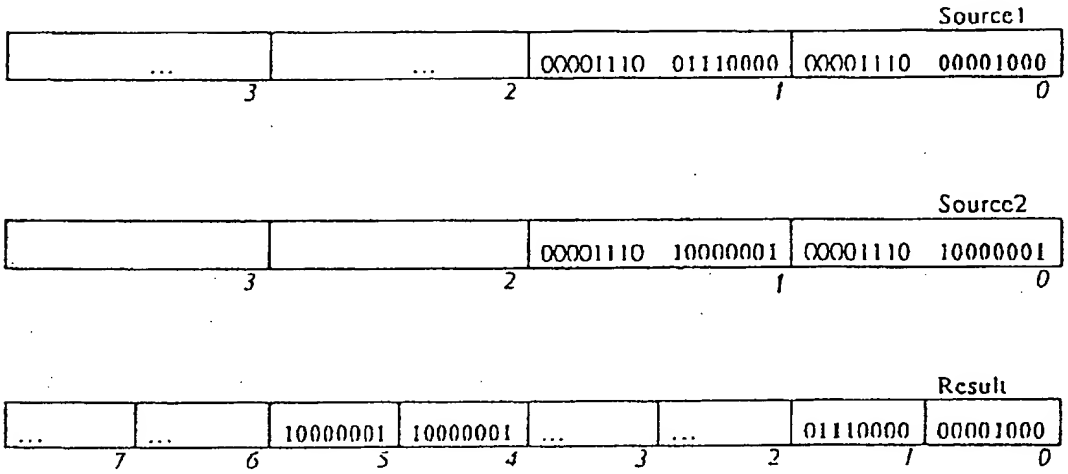
る。８ビットの出力は入力の下位８ビットか、または任意選択によりクランプ値（０×８０、０×７Ｆ、または０×ＦＦ）である。飽和検査回路８１０は、Ｓｏｕｒｃｅ１のビット１５ないし０を受け取り、結果レジスタ８５２に対するビット７ないし０を出力する。飽和検査回路８１１は、Ｓｏｕｒｃｅ１のビット３１ないし１６を受け取り、結果レジスタ８５２に対するビット１５ないし８を出力する。飽和検査回路８１２は、Ｓｏｕｒｃｅ１のビット４７ないし３２を受け取り、結果レジスタ８５２に対するビット２３ないし１６を出力する。飽和検査回路８１３は、Ｓｏｕｒｃｅ１のビット６３ないし４８を受け取り、結果レジスタ８５２に対するビット３１ないし２４を出力する。飽和検査回路８１４は、Ｓｏｕｒｃｅ２のビット１５ないし０を受け取り、結果レジスタ８５２に対するビット３９ないし３２を出力する。飽和検査回路８１５は、Ｓｏｕｒｃｅ２のビット３１ないし１６を受け取り、結果レジスタ８５２に対するビット４７ないし４０を出力する。飽和検査回路８１６は、Ｓｏｕｒｃｅ２のビット４７ないし３２を受け取り、結果レジスタ８５２に対するビット５５ないし４８を出力する。飽和検査回路８１７は、Ｓｏｕｒｃｅ２のビット６３ないし４８を受け取り、結果レジスタ８５２に対するビット６３ないし５６を出力する。

飽和検査回路８２０ないし飽和検査回路８２３は、３２ビットの入力と１６ビットの出力を有する。１６ビットの出力は、入力の下位１６ビットか、または任意選択によりクランプ値（０×８０００、０×７ＦＦＦ、または０×ＦＦＦＦ）である。飽和検査回路８２０は、Ｓｏｕｒｃｅ１のビット３１ないしゼロを受け取り、結果レジスタ８５３に対するビット１５ないし０を出力する。飽和検査回路８２１がＳｏｕｒｃｅ１のビット６３ないし３２を受け取り、結果レジスタ８５３に対するビット３１ないし１６を出力する。飽和検査回路８２２は、Ｓｏｕｒｃｅ２のビット３１ないし０を受け取り、結果レジスタ８５３に対するビット４７ないし３２を出力する。飽和検査回路８２３は、Ｓｏｕｒｃｅ２のビット６３ないし３２を受け取り、結果レジスタ８５３に対するビット６３ないし４８を出力する。

たとえば、表４に飽和を使用しない無符号パックドワードを示す。演算制御回

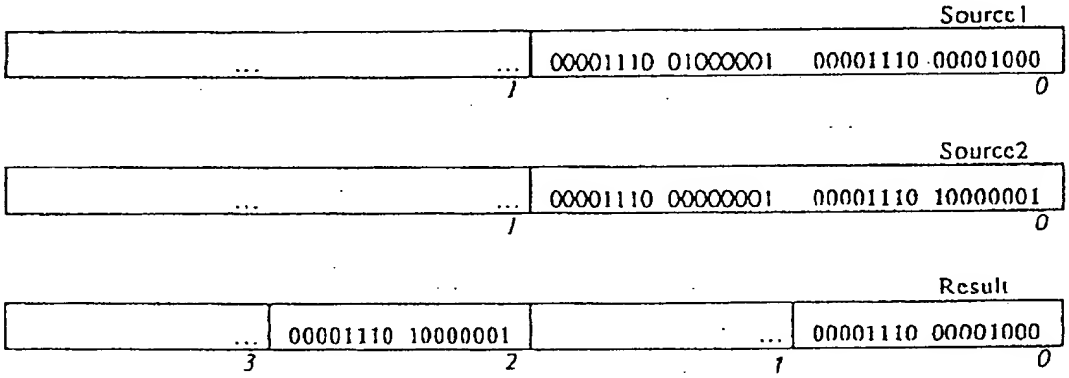
路８００は、結果レジスタ８５２が結果［６３：０］８６０を出力することができるようにする。

表４



しかし、飽和なしの無符号パックドダブルワード演算を行う場合、演算制御回路８００は結果レジスタ８５３が結果［６３：０］８６０を出力することができるようにする。表５にこの結果を示す。

表５



アンパック演算

一実施例では、アンパック演算によって、２つのソース・パックドデータの下位パックドバイト、ワード、またはダブルワードをインタリーブして、結果のパックドバイト、ワード、またはダブルワードを生成する。

第９図に、パックドデータに対するアンパック演算を行う方法の一実施例を示す。この実施例は、第２図のプロセッサ１０９において実施することができる。

先にステップ７０１およびステップ７０２が実行される。ステップ９０３で、デコーダ２０２がアンパック演算を行うように機能ユニット２０３を使用可能にする。デコーダ２０２は内部バス２０５を介してＳｏｕｒｃｅ１およびＳｏｕｒｃｅ２内のデータ要素のサイズを伝達する。

ステップ９１０で、このデータ要素のサイズによって次にどのステップを実行するかが決まる。データ要素のサイズが８ビット（パックドバイト５０１データ）の場合、機能ユニット２０３はステップ７１２を実行する。しかし、パックドデータ内のデータ要素のサイズが１６ビット（パックドワード５０２データ）の場合、機能ユニット２０３はステップ７１４を実行する。しかし、パックドデータ内のデータ要素のサイズが３２ビット（パックドダブルワード５０３データ）の場合、機能ユニット２０３はステップ７１６を実行する。

ソース・データ要素のサイズが８ビットであると仮定すると、ステップ７１２が実行される。ステップ７１２では以下のように行われる。Ｓｏｕｒｃｅ１のビット７～０はＲｅｓｕｌｔのビット７～０である。Ｓｏｕｒｃｅ２のビット７～０はＲｅｓｕｌｔのビット１５～８である。Ｓｏｕｒｃｅ１のビット１５～８はＲｅｓｕｌｔのビット２３～１６である。Ｓｏｕｒｃｅ２のビット１５～８はＲｅｓｕｌｔのビット３１～２４である。Ｓｏｕｒｃｅ１のビット２３～１６はＲｅｓｕｌｔのビット３９～２２である。Ｓｏｕｒｃｅ２のビット２３～１６はＲｅｓｕｌｔのビット４７～４０である。Ｓｏｕｒｃｅ１のビット３１～２４はＲｅｓｕｌｔのビット５５～４８である。Ｓｏｕｒｃｅ２のビット３１～２４はＲｅｓｕｌｔのビット６３～５６である。

ソース・データ要素のサイズが１６ビットであると仮定すると、ステップ７１４が実行される。ステップ７１４では以下のように行われる。Ｓｏｕｒｃｅ１のビット１５～０はＲｅｓｕｌｔのビット１５～０である。Ｓｏｕｒｃｅ２のビット１５～０はＲｅｓｕｌｔのビット３１～１６である。Ｓｏｕｒｃｅ１のビット３１～１６はＲｅｓｕｌｔのビット４７～３２である。Ｓｏｕｒｃｅ２のビット３１～１６はＲｅｓｕｌｔのビット６３～４８である。

ソース・データ要素のサイズが３２ビットであると仮定すると、ステップ７１

６が実行される。ステップ７１６では以下のように行われる。Ｓｏｕｒｃｅ１のビット３１～０がＲｅｓｕｌｔのビット３１～０である。Ｓｏｕｒｃｅ２のビット３１～０がＲｅｓｕｌｔのビット６３～３２である。

一実施例では、ステップ７１２のアンパック演算が同時に行われる。しかし、他の実施例では、このアンパック演算は順次に行われる。他の実施例では、このアンパック演算の一部が同時に行われ、一部は順次に行われる。これは、ステップ７１４および７１６のアンパック演算にも適用される。

ステップ７２０で、ＲｅｓｕｌｔがＤＥＳＴ６０５レジスタに格納される。

表６に、アンパック・バイト演算のレジスタ内表現を示す。

表 6

Source1

00101010	01010101	01010101	11111111	10000000	01110000	10001111	10001000
7	6	5	4	3	2	1	0

Source2

00000000	00000000	11000000	00000000	11110011	00000000	10001110	10001000
7	6	5	4	3	2	1	0

Result

11110011	10000000	00000000	01110000	10001110	10001111	10001000	10001000
7	6	5	4	3	2	1	0

表７に、アンパック・ワード演算のレジスタ内表現を示す。

表 7

Source1

00101010 01010101	01010101 11111111	10000000 01110000	10001111 10001000
3	2	1	0

Source2

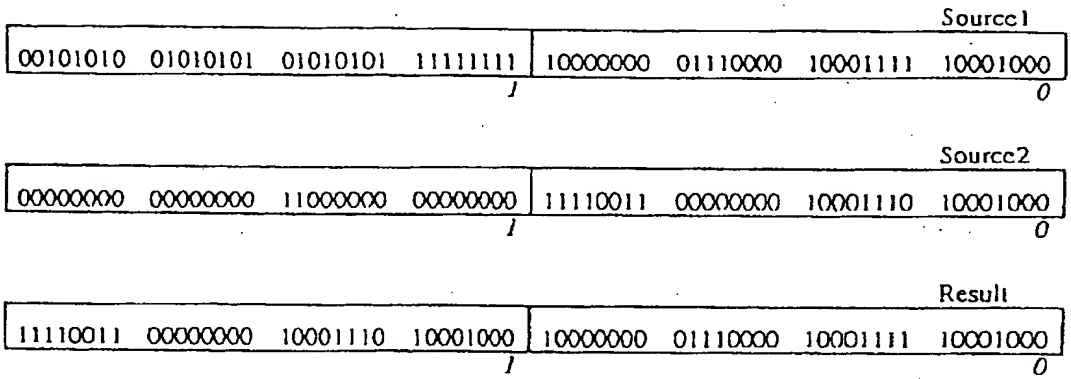
00000000 00000000	11000000 00000000	11110011 00000000	10001110 10001000
3	2	1	0

Result

11110011 00000000	10000000 01110000	10001110 10001000	10001111 10001000
3	2	1	0

表８に、アンパック・ダブルワード演算のレジスタ内表現を示す。

表８



アンパック回路

本発明の一実施例では、アンパック演算の効率的な実行を実現するために並列処理を使用する。第１０図に、パックドデータに対するアンパック演算を実行することができる回路の一実施例を示す。

第１０図の回路は、演算制御回路８００と結果レジスタ１０５２と結果レジスタ１０５３と結果レジスタ１０５４とを備える。

演算制御回路８００は、デコーダ２０２からアンパック演算をイネーブルにする情報を受け取る。ソース・パックドデータのサイズがバイト・パックドデータ５０２の場合、演算制御回路８００によって出力イネーブル１０３２が設定される。これによって、結果レジスタ１０５２の出力がイネーブルにされる。ソース・パックドデータのサイズがワード・パックドデータ５０３の場合、演算制御回路８００によって出力イネーブル１０３３が設定される。これによって出力レジスタ１０５３の出力がイネーブルにされる。ソース・パックドデータのサイズがダブルワード・パックドデータ５０４の場合、演算制御回路８００によって出力イネーブル１０３４が設定される。これによって、出力結果レジスタ１０５４の出力がイネーブルにされる。

結果レジスタ１０５２は以下の入力を有する。Ｓｏｕｒｃｅ１のビット７～０は結果レジスタ１０５２のビット７～０である。Ｓｏｕｒｃｅ２のビット７～０は結果レジスタ１０５２のビット１５～８である。Ｓｏｕｒｃｅ１のビット１５～８は結果レジスタ１０５２のビット２３～１６である。Ｓｏｕｒｃｅ２のビッ

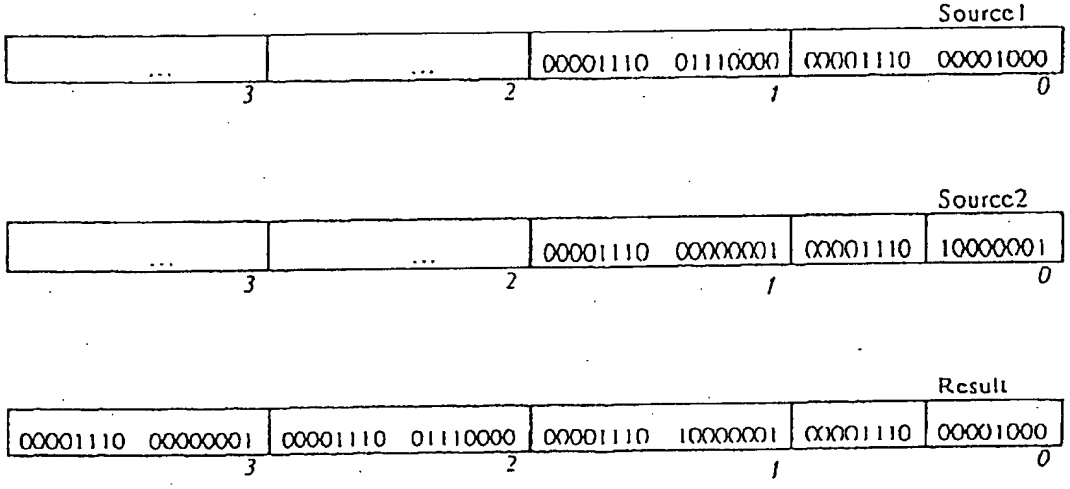
ト１５～８は結果レジスタ１０５２のビット３１～２４である。Ｓｏｕｒｃｅ１のビット２３～１６は結果レジスタ１０５２のビット３９～３２である。Ｓｏｕｒｃｅ２のビット２３～１６は結果レジスタ１０５２のビット４７～４０である。Ｓｏｕｒｃｅ１のビット３１～２４は結果レジスタ１０５２の５５～４８である。Ｓｏｕｒｃｅ２のビット３１～２４は結果レジスタ１０５２のビット６３～５６である。

結果レジスタ１０５３は以下の入力を有する。Ｓｏｕｒｃｅ１のビット１５～０は結果レジスタ１０５３の１５～０である。Ｓｏｕｒｃｅ２のビット１５～０は結果レジスタ１０５３の３１～１６である。Ｓｏｕｒｃｅ１のビット３１～１６は結果レジスタ１０５３のビット４７～３２である。Ｓｏｕｒｃｅ２のビット３１～１６は結果レジスタ８５３のビット６３～４８である。

結果レジスタ１０５４は以下の入力を有する。Ｓｏｕｒｃｅ１のビット３１～０は結果レジスタ１０５４のビット３１～０である。Ｓｏｕｒｃｅ２のビット３１～０は結果レジスタ１０５４のビット６３～３２である。

たとえば、表９ではアンパック・ワード演算が実行される。演算制御回路８００は結果レジスタ１０５３が結果［６３：０］８６０を出力することができるようにする。

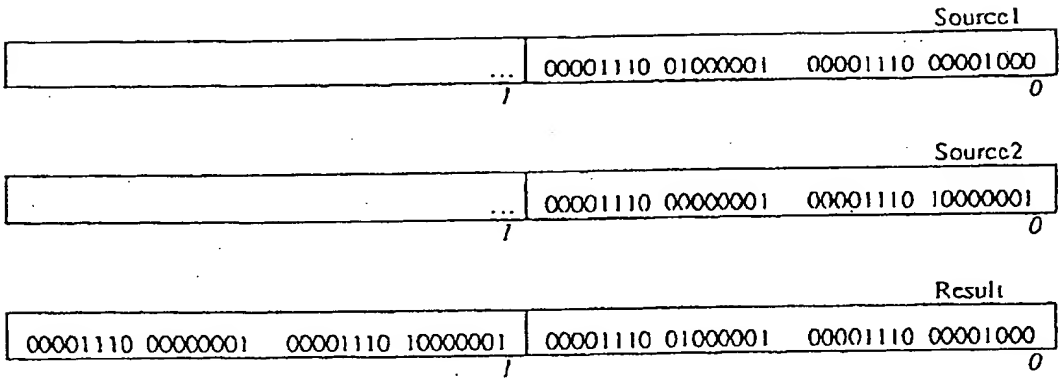
表９



しかし、アンパック・ダブルワード演算を行う場合、演算制御回路８００は結果

レジスタ１０５４が結果［６３：０］８６０を出力することができるようにする。表１０にその結果を示す。

表１０



したがって、移動演算、パック演算、およびアンパック演算によってマルチメディア・データ要素を操作することができる。従来技術のプロセッサでは、これらのタイプの操作を行うために、単一のパック移動演算、パック演算、またはアンパック演算を行うために複数の別々の演算を実行する必要がある。一実施例では、パックドデータ演算のためのデータ線はすべて関係のあるデータを伝送する。これによって、コンピュータ・システムのパフォーマンスが向上する。

【図１】

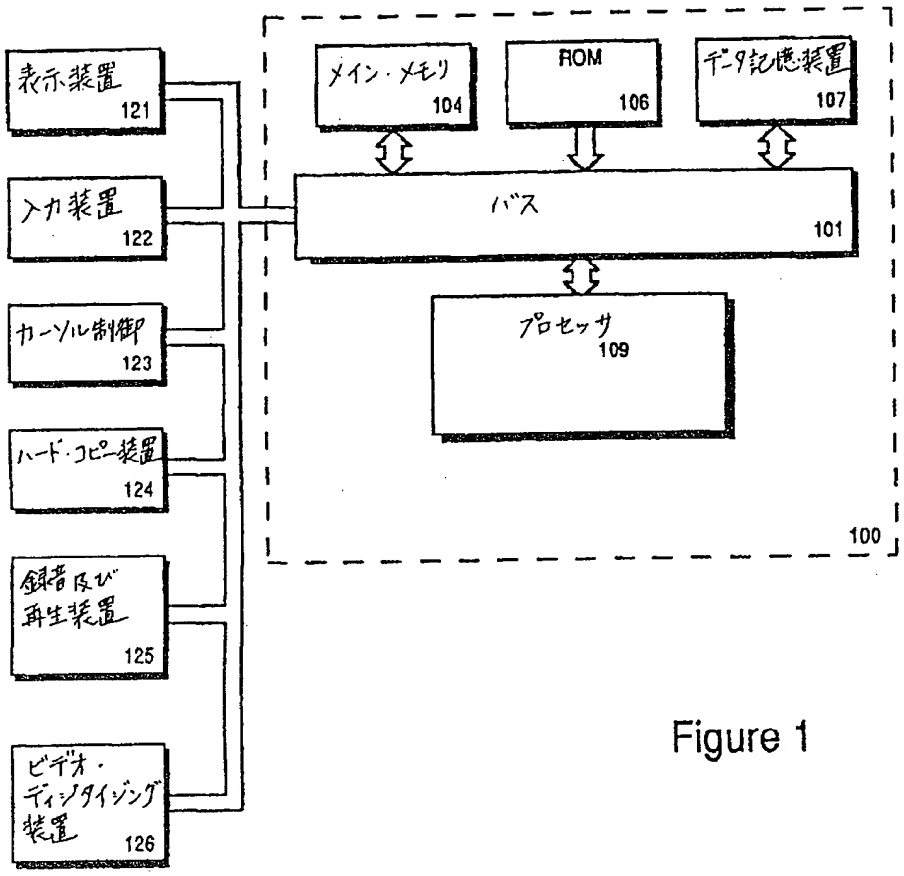

表示装置
121
入力装置
122
カーソル制御
123
ハード・コピー装置
124
録音及び
再生装置
125
ビデオ・
ディジタイジング
装置
126
メイン・メモリ
104
ROM
106
データ記憶装置
107
バス
101
プロセッサ
109
100

Figure 1

【図２】

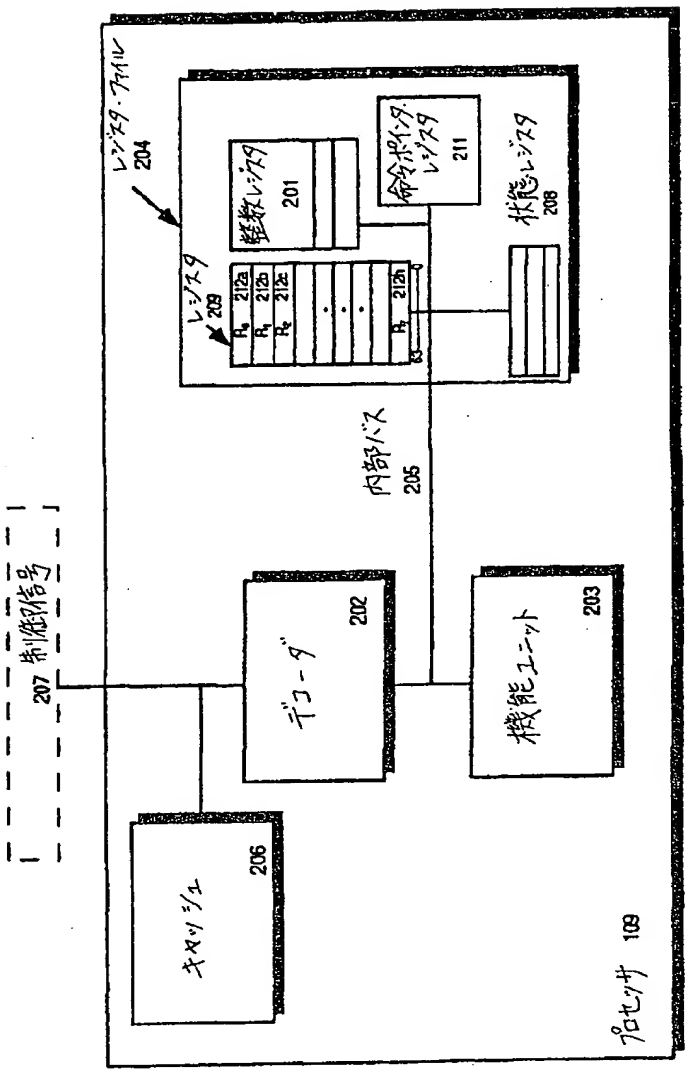

レジスタ・ファイル
204
整数レジスタ
201
命令ポインタ・レジスタ
211
状態レジスタ
208
レジスタ
209
R0 212a
R1 212b
R2 212c
Rn 212n
63
内部バス
205
制御信号
207
デコーダ
202
機能ユニット
203
マルチプレクサ
206
プロセッサ
109

Figure 2

【図３】

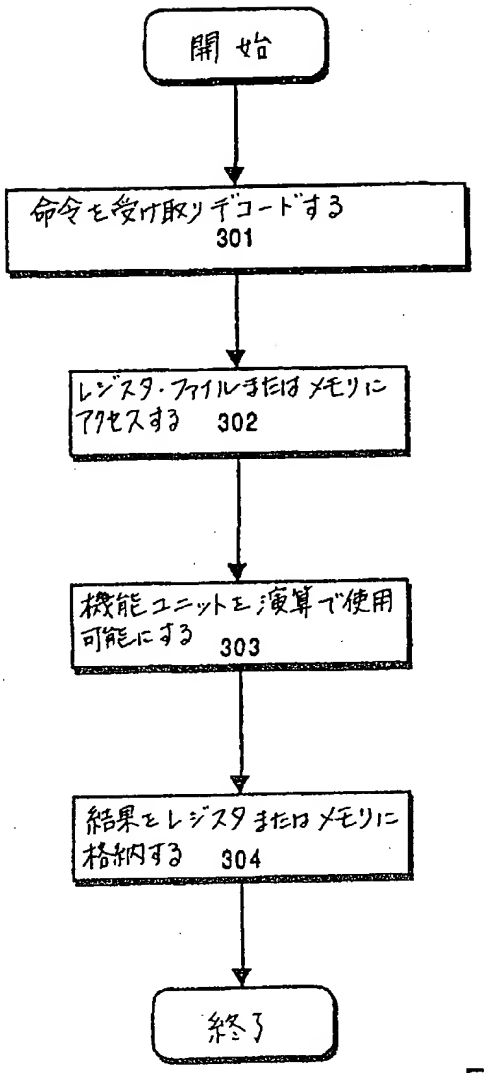

開始
命令を受け取りデコードする
301
レジスタ・ファイルまたはメモリに
アクセスする 302
機能ユニットを演算で使用
可能にする 303
結果をレジスタまたはメモリに
格納する 304
終了

Figure 3

【図４】

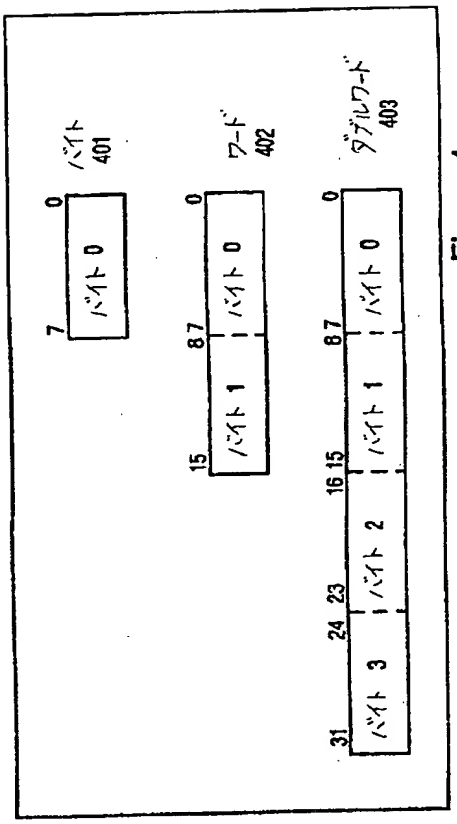

バイト
401
ワード
402
ダブルワード
403
バイト 0
バイト 1
バイト 2
バイト 3
0
7
8
15
16
23
24
31

Figure 4a

【図４】

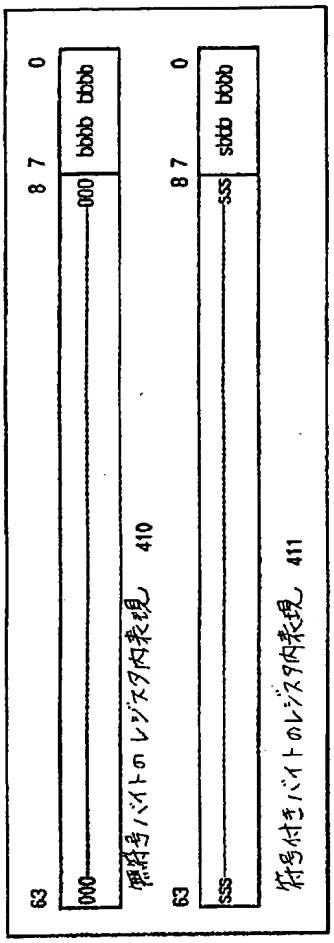

63
8 7
0
000
bbbb bbbb
無符号バイトのレジスタ内表現
410
63
8 7
0
sss
sbbb bbbb
符号付きバイトのレジスタ内表現
411

Figure 4b

【図４】

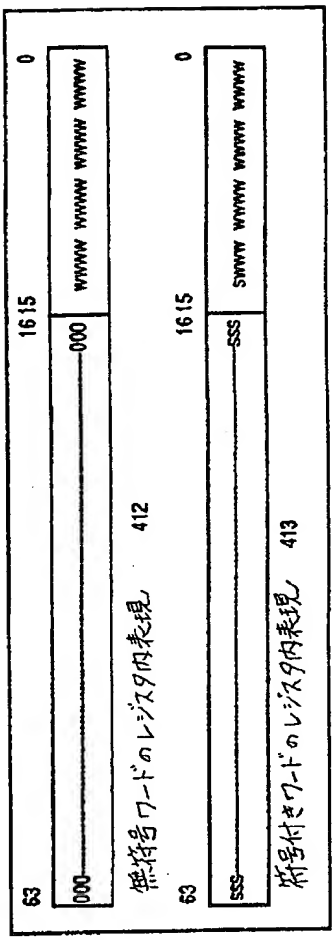

63
16 15
0
000
wwww wwww wwww wwww
000
無符号ワードのレジスタ内表現
412
63
16 15
0
sss
swww wwww wwww wwww
sss
符号付きワードのレジスタ内表現
413

Figure 4c

【図４】

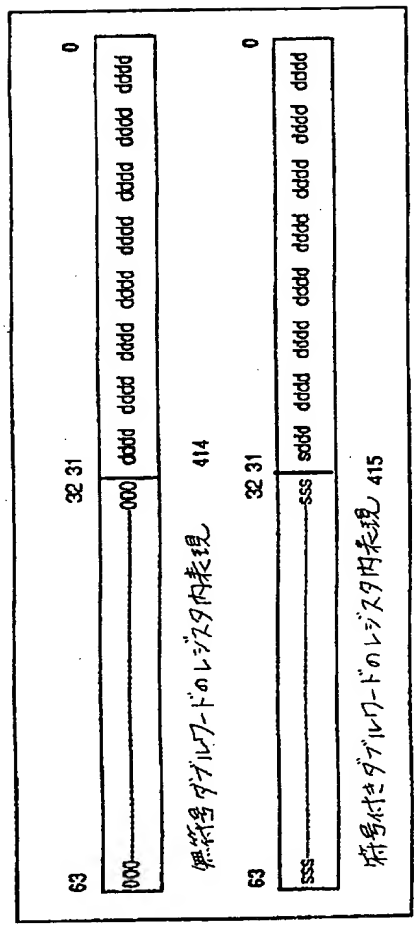

63
32 31
0
dddd dddd dddd dddd dddd dddd dddd dddd
000 000
無符号ダブルワードのレジスタ内表現 414
63
32 31
0
sddd dddd dddd dddd dddd dddd dddd dddd
sss sss
符号付きダブルワードのレジスタ内表現 415

Figure 4d

【図５】

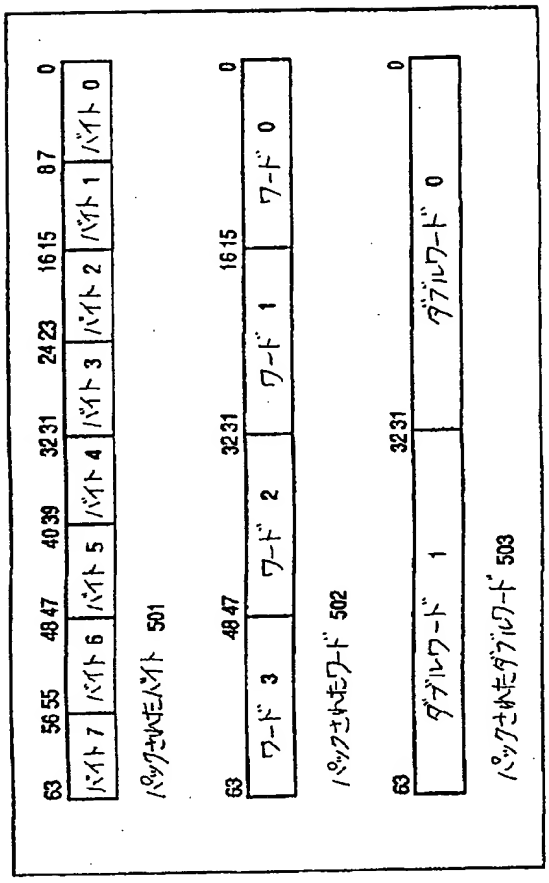

63 56 55 48 47 40 39 32 31 24 23 16 15 8 7 0
バイト 7 バイト 6 バイト 5 バイト 4 バイト 3 バイト 2 バイト 1 バイト 0
パックされたバイト 501
63 48 47 32 31 16 15 0
ワード 3 ワード 2 ワード 1 ワード 0
パックされたワード 502
63 32 31 0
ダブルワード 1 ダブルワード 0
パックされたダブルワード 503

Figure 5a

【図５】

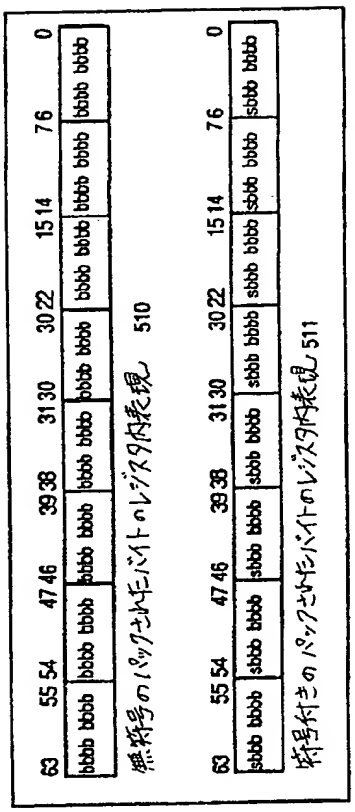

63 55 54 47 46 39 38 31 30 22 15 14 7 6 0
bbbb bbbb bbbb bbbb bbbb bbbb bbbb bbbb bbbb bbbb bbbb bbbb bbbb bbbb bbbb bbbb
無符号のパックされたバイトのレジスタ内表現 510
63 55 54 47 46 39 38 31 30 22 15 14 7 6 0
sbbb bbbb sbbb bbbb sbbb bbbb sbbb bbbb sbbb bbbb sbbb bbbb sbbb bbbb sbbb bbbb
符号付きのパックされたバイトのレジスタ内表現 511

Figure 5b

【図５】

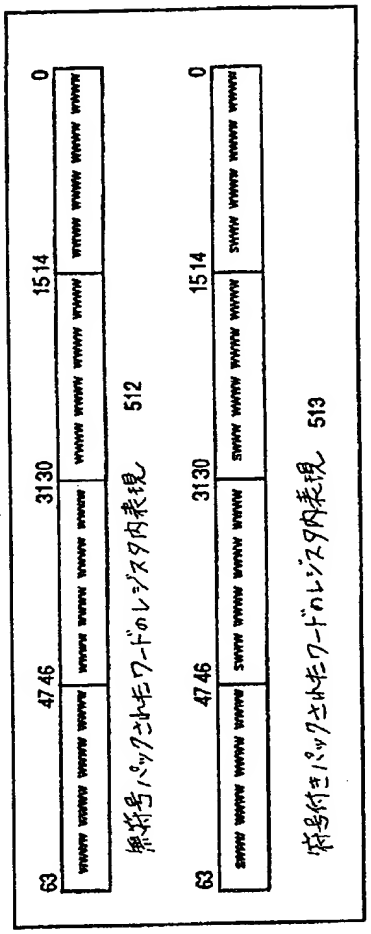

63
47 46
3130
1514
0
無符号パックされワードのレジスタ内表現　512
63
47 46
3130
1514
0
符号付きパックされワードのレジスタ内表現　513

Figure 5c

【図５】

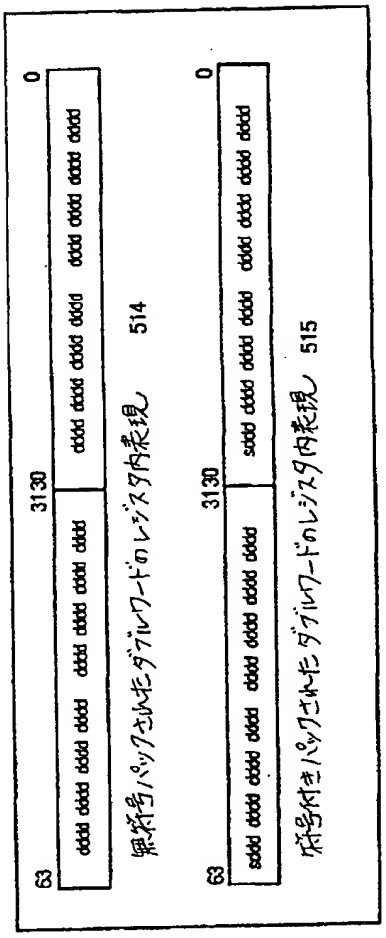


Figure 5d

【図６】

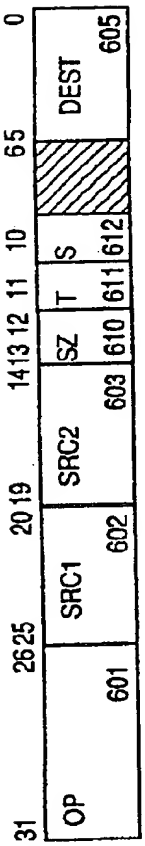


Figure 6a

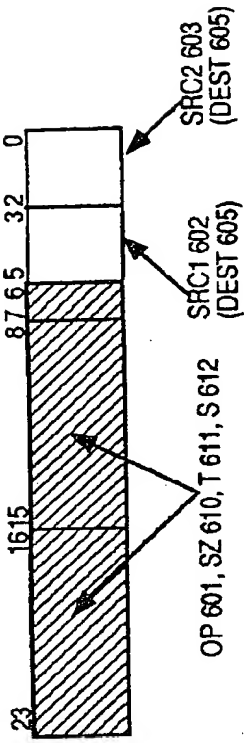


Figure 6b

【図７】

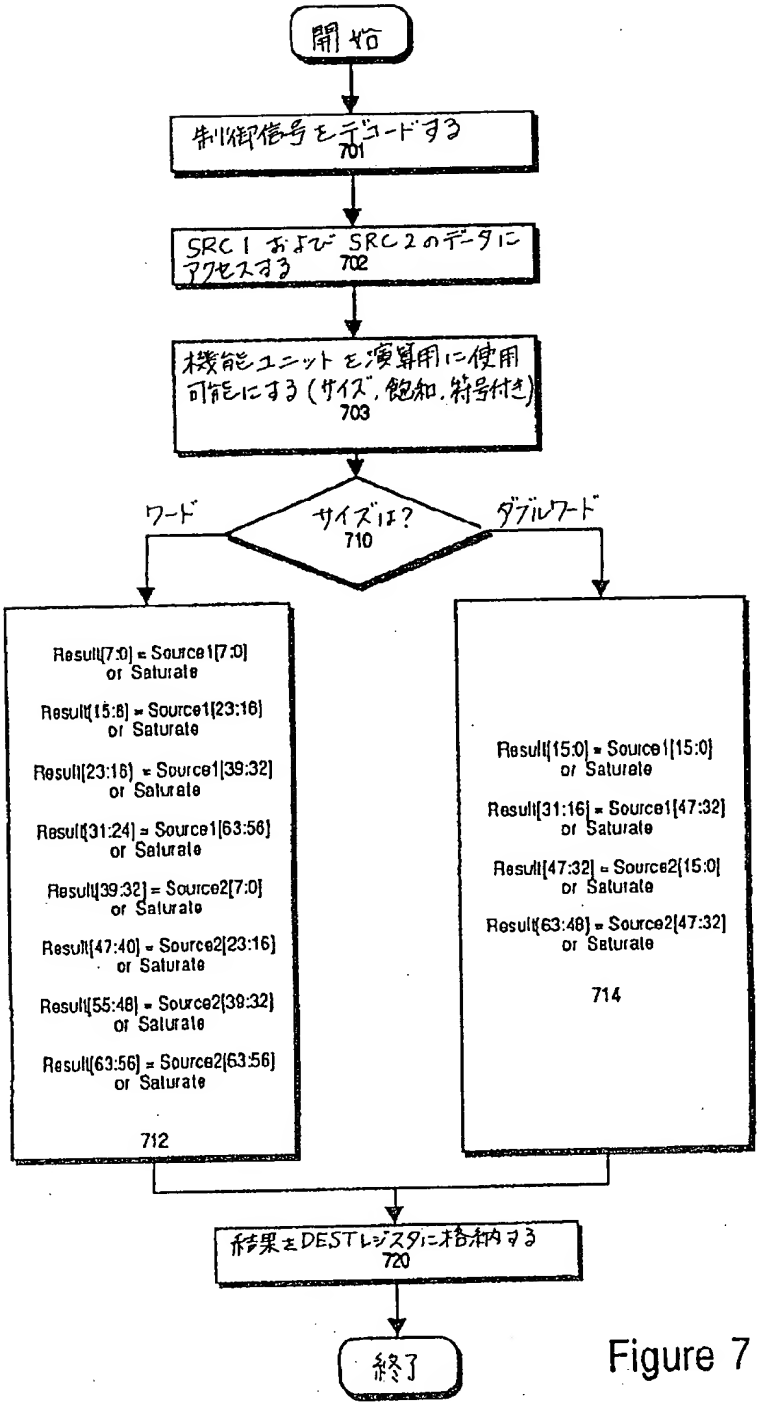

開始
制御信号をデコードする
701
SRC1 および SRC2のデータにアクセスする
702
機能ユニットを演算用に使用可能にする（サイズ，飽和，符号付き）
703
サイズは？
710
ワード
ダブルワード
Result[7:0] = Source1[7:0] or Saturate
Result[15:8] = Source1[23:16] or Saturate
Result[23:16] = Source1[39:32] or Saturate
Result[31:24] = Source1[63:56] or Saturate
Result[39:32] = Source2[7:0] or Saturate
Result[47:40] = Source2[23:16] or Saturate
Result[55:48] = Source2[39:32] or Saturate
Result[63:56] = Source2[63:56] or Saturate
712
Result[15:0] = Source1[15:0] or Saturate
Result[31:16] = Source1[47:32] or Saturate
Result[47:32] = Source2[15:0] or Saturate
Result[63:48] = Source2[47:32] or Saturate
714
結果をDESTレジスタに格納する
720
終了

Figure 7

【図８】

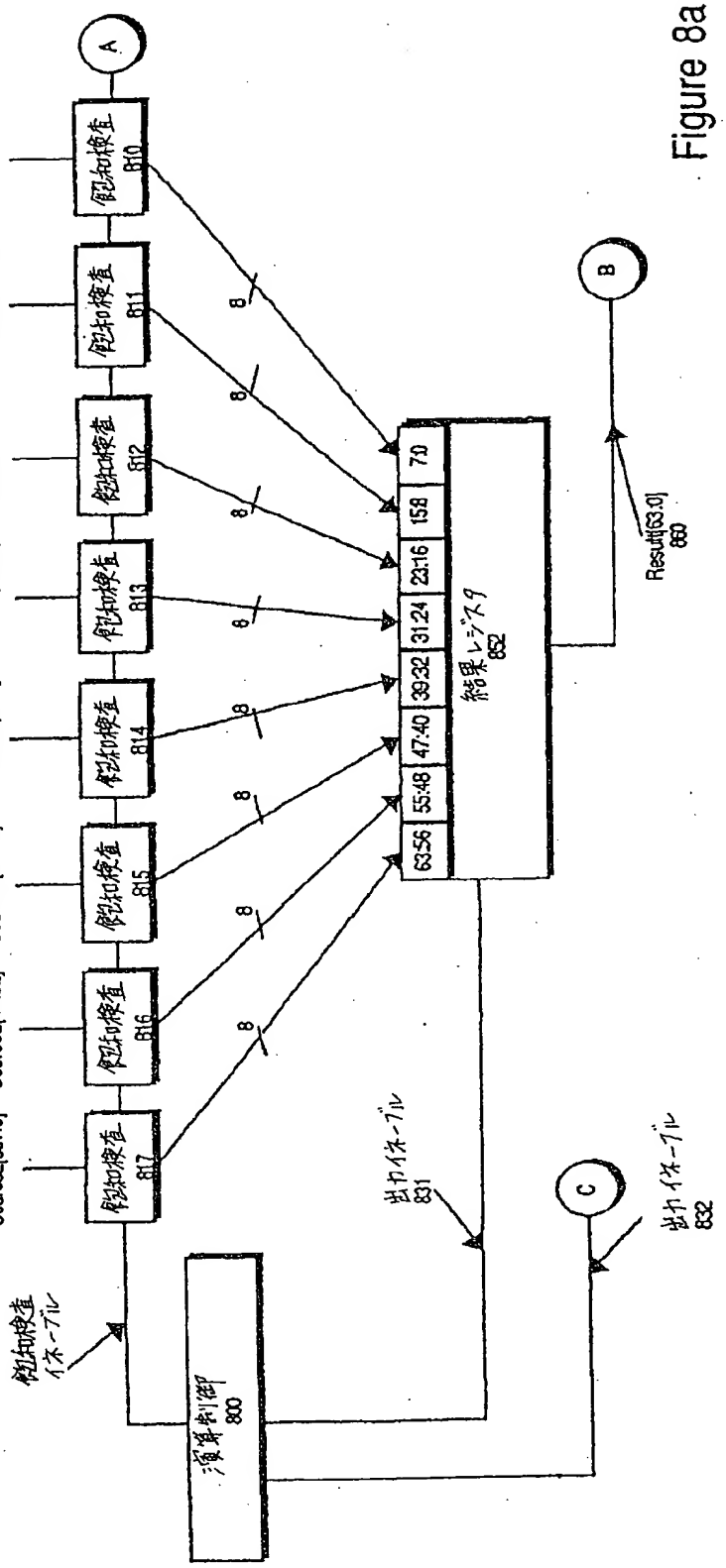

A
B
C
Source1[15:0]
Source1[31:16]
Source1[47:32]
Source1[63:48]
Source2[15:0]
Source2[31:16]
Source2[47:32]
Source2[63:48]
例知検査 810
例知検査 811
例知検査 812
例知検査 813
例知検査 814
例知検査 815
例知検査 816
例知検査 817
8
例知検査イネーブル
演算制御 800
出力イネーブル 831
出力イネーブル 832
結果レジスタ 852
7:0
15:8
23:16
31:24
39:32
47:40
55:48
63:56
Result[63:0] 860

Figure 8a

【図８】

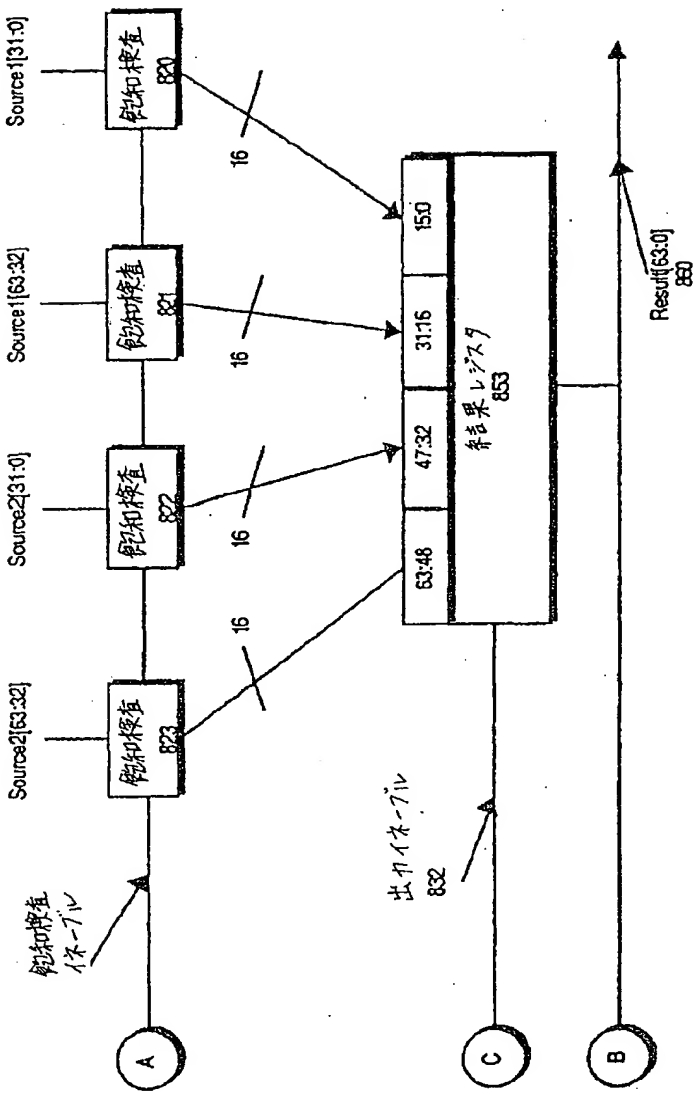

Source1[31:0]
Source1[63:32]
Source2[31:0]
Source2[63:32]
検知検査 820
検知検査 821
検知検査 822
検知検査 823
16
16
16
16
63:48
47:32
31:16
15:0
結果レジスタ 853
Result[63:0] 860
出力イネーブル 832
検知検査イネーブル
A
B
C

Figure 8b

【図９】

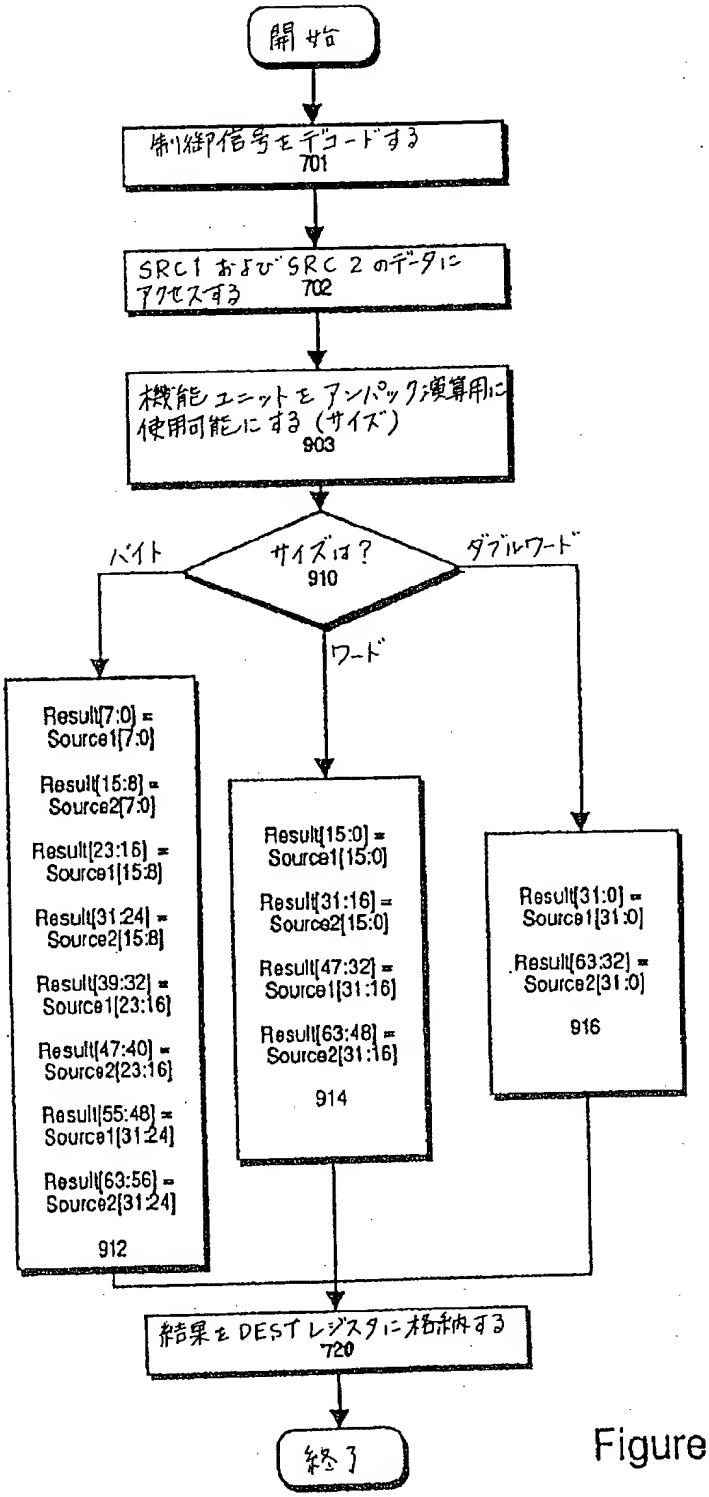

開始
制御信号をデコードする
701
SRC1 および SRC 2 のデータに
アクセスする　702
機能ユニットをアンパック演算用に
使用可能にする（サイズ）
903
サイズは？
910
バイト
ワード
ダブルワード
Result[7:0] =
Source1[7:0]
Result[15:8] =
Source2[7:0]
Result[23:16] =
Source1[15:8]
Result[31:24] =
Source2[15:8]
Result[39:32] =
Source1[23:16]
Result[47:40] =
Source2[23:16]
Result[55:48] =
Source1[31:24]
Result[63:56] =
Source2[31:24]
912
Result[15:0] =
Source1[15:0]
Result[31:16] =
Source2[15:0]
Result[47:32] =
Source1[31:16]
Result[63:48] =
Source2[31:16]
914
Result[31:0] =
Source1[31:0]
Result[63:32] =
Source2[31:0]
916
結果を DEST レジスタに格納する
720
終了

Figure 9

【図１０】

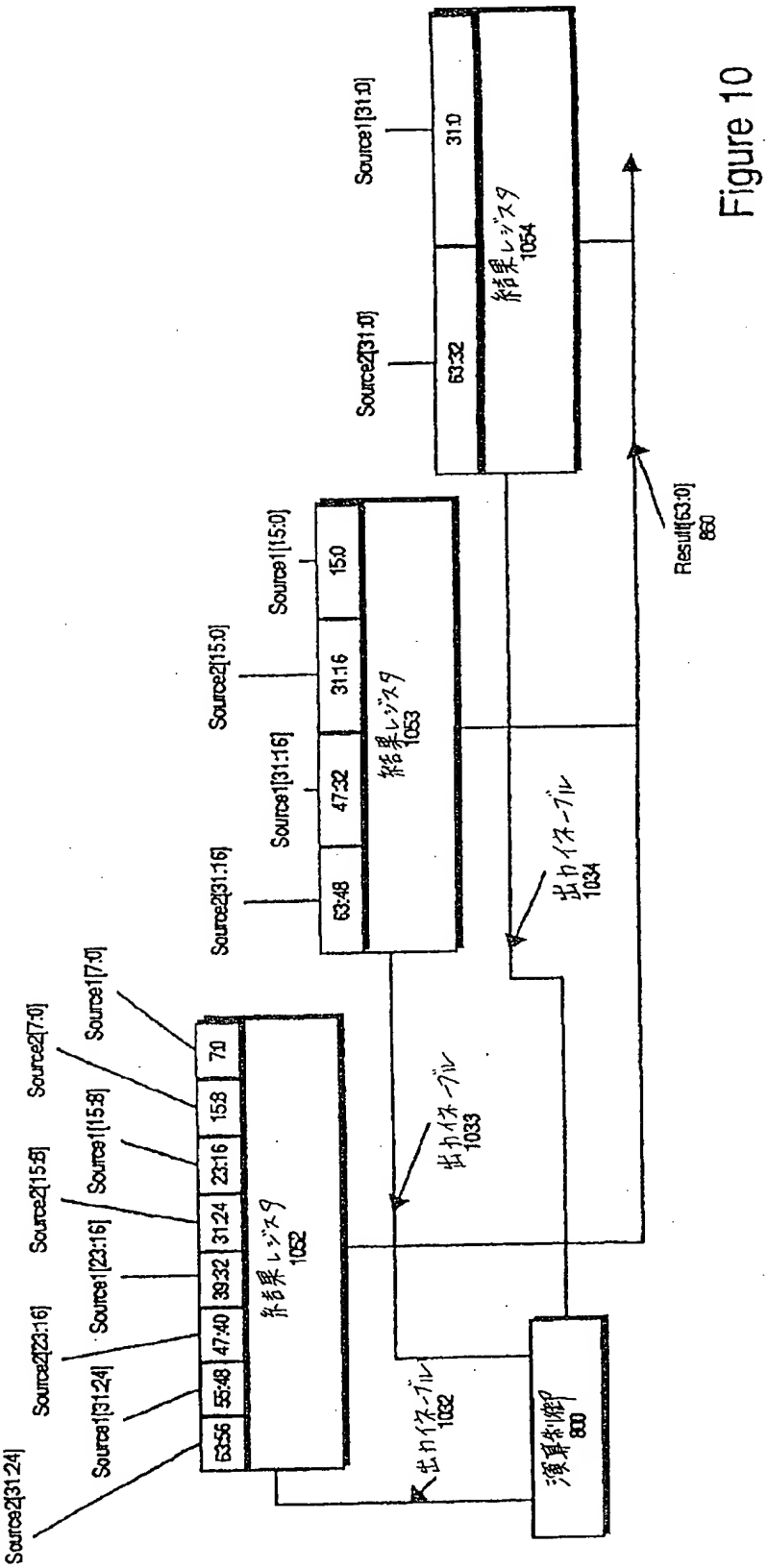


Figure 10

【手続補正書】特許法第１８４条の８第１項

【提出日】１９９６年９月２４日

【補正内容】

補正請求の範囲

１．第１の複数のデータ要素および第２の複数のデータ要素をそれぞれ含む第１のパックドデータおよび第２のパックドデータにして、前記第１の複数のデータ要素内の各データ要素が前記第２の複数のデータ要素内の異なるデータ要素に対応している、第１のパックドデータと第２のパックドデータを保持するよう構成された記憶域と、

アンパック命令をデコードするように構成されたデコーダと、

前記記憶域と前記デコーダとに結合され、アンパック命令に応答して、前記第１および第２の複数のデータ要素から選択された対応するデータ要素を第３のパックドデータ内の第３の複数のデータ要素として前記第１の記憶域に同時にコピーするように構成された回路と

を備えるプロセッサ。

２．前記第１の複数のデータ要素からコピーされるデータ要素と前記第２の複数のデータ要素からコピーされるデータ要素が前記第３の複数のデータ要素として前記記憶域にインタリーブされることを特徴とする、請求項１に記載のプロセッサ。

３．前記回路が前記第１および第２の複数のデータ要素内のデータ要素の半分を同時にコピーするように構成された、請求項２に記載のプロセッサ。

４．前記第１の複数のデータ要素および前記第２の複数のデータ要素がそれぞれ２個、４個、または８個のデータ要素を含むことを特徴とする、請求項１に記載のプロセッサ。

５．コピーされて前記第３の複数のデータ要素を生成する前記第１の複数のデータ要素のそれぞれのデータ要素が、前記第１の複数のデータ要素が前記第１のパックドデータ内で現れる順序と同じ順序で格納されることを特徴とする、請求項２に記載のプロセッサ。

６．第１の複数のデータ要素および第２の複数のデータ要素をそれぞれ含む第

１のパックドデータおよび第２のパックドデータにして、前記第１の複数のデータ要素内の各データ要素が前記第２の複数のデータ要素内の異なるデータ要素に

対応している、第１のパックドデータと第２のパックドデータを保持するよう構成された記憶域と、

パック命令をデコードするように構成されたデコーダと、

前記記憶域と前記デコーダとに結合されパック命令に応答して、前記第１および第２の複数のデータ要素内の各データ要素の一部を第３のパックドデータ内の第３の複数のデータ要素として前記記憶域に同時にコピーするように構成された回路と

を備えるプロセッサ。

７．前記一部が前記第１および第２の複数のデータ要素内の各データ要素内のビットの半分であることを特徴とする、請求項６に記載のプロセッサ。

８．前記一部が前記第１および第２の複数のデータ要素内の各データ要素の下位ビットまたは上位ビットであることを特徴とする、請求項７に記載のプロセッサ。

９．前記第１の複数のデータ要素内のデータ要素からコピーされる一部が、前記第３の複数のデータ要素内で隣接して格納されることを特徴とする、請求項８に記載のプロセッサ。

１０．前記第１および第２の複数のデータ要素からコピーされる前記一部が、前記第１および第２の複数のデータ要素が前記第１および第２のパックドデータ内で現れるのと同じ順序で格納されることを特徴とする、請求項９に記載のプロセッサ。

１１．前記第１および第２の複数のデータ要素内のすべてのデータ要素が符号付きで、前記第３の複数のデータ要素内のすべてのデータ要素が符号付きまたは無符号であることを特徴とする、請求項１０に記載のプロセッサ。

１２．前記第３の複数のデータ要素のすべてのデータ要素が飽和または非飽和であることを特徴とする、請求項１１に記載のプロセッサ。

１３．前記第１の複数のデータ要素と前記第２の複数のデータ要素がそれぞれ

２個、４個、または８個のデータ要素を含むことを特徴とする、請求項６に記載のプロセッサ。

【国際調査報告】

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US95/15713

A. CLASSIFICATION OF SUBJECT MATTER
IPC(6) :G06F 7/14, 9/30
US CL :395/375, 800
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED
Minimum documentation searched (classification system followed by classification symbols)
U.S. : 395/375, 800

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
APS

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X,P	US, A, 5,423,010 (MIZUKAMI) 06 JUNE 1995, FIGS. 2A, 12 AND 13, COL. 3	1-14
X,P	US, A, 5,408,670 (DAVIES) 18 APRIL 1995, COL. 6, LINES 21-29	1-14
Y	US, A, 5,168,571 (HOOVER ET AL) 01 DECEMBER 1992, FIGS. 4 AND 5, COL. 6	1-14
Y	US, A, 4,139,899 (TULPULE ET AL) 13 FEBRUARY 1979, ENTIRE DOCUMENT	1-14
Y,P	US, A, 5,465,374 (DINKJIAN ET AL) 07 NOVEMBER 1995, ENTIRE DOCUMENT	1-14
A	US, A, 4,595,911 (KREGNESS ET AL) 17 JUNE 1986	1-14

[X] Further documents are listed in the continuation of Box C. [ ] See patent family annex.

*	Special categories of cited documents:	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A"	document defining the general state of the art which is not considered to be part of particular relevance	"X"	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E"	earlier document published on or after the international filing date	"Y"	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L"	document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"&"	document member of the same patent family
"O"	document referring to an oral disclosure, use, exhibition or other means		
"P"	document published prior to the international filing date but later than the priority date claimed		

Date of the actual completion of the international search
27 FEBRUARY 1996

Date of mailing of the international search report
12 MAR 1996

Name and mailing address of the ISA/US
Commissioner of Patents and Trademarks
Box PCT
Washington, D.C. 20231
Facsimile No. (703) 305-3230

Authorized officer
KENNETH S. KIM
Telephone No. (703) 305-9693

Form PCT/ISA/210 (second sheet)(July 1992)*

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US95/15713

C (Continuation): DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US, A, 5,276,891 (PATEL) 04 JANUARY 1994	1-14
A	US, A, 5,265,204 (KIMURA ET AL) 23 NOVEMBER 1993	1-14
A	US, A, 5,327,543 (MIURA ET AL) 05 JULY 1994	1-14
A,P	US, A, 5,390,135 (LEE ET AL) 14 FEBRUARY 1995	1-14

フロントページの続き

(81)指定国 EP(AT, BE, CH, DE, DK, ES, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OA(BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG), AP(KE, LS, MW, SD, SZ, UG), AL, AM, AT, AT, AU, BB, BG, BR, BY, CA, CH, CN, CZ, CZ, DE, DE, DK, DK, EE, EE, ES, FI, FI, GB, GE, HU, IS, JP, KE, KG, KP, KR, KZ, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SK, TJ, TM, TT, UA, UG, UZ, VN

(72)発明者 ミタル，ミリンド
アメリカ合衆国・94080・カリフォルニア州・サウス サンフランシスコ・ヒルサイド ブルバード・1149

(72)発明者 メネマイアー，ラリー・エム
アメリカ合衆国・95006・カリフォルニア州・ボルダー クリーク・ピーオー ボックス・587

(72)発明者 エイタン，ベニー
イスラエル国・ハイファ・スティーブンウィース・25